6. Robot calibration

Note: The parameters have been calibrated before the product leaves the factory, and calibration is generally not required. If you feel that the control robot deviates greatly, you need to calibrate [imu], [linear speed], and [angular speed]; when calibrating, place the robot in advance and do not move the robot.

6.1, imu calibration

The error of IMU mainly comes from three parts, including noise (Bias and Noise), scale factor (Scale errors) and axis deviation (Axis misalignments).

Calculate accelerometer calibration parameters. Because it requires keyboard input, it should be run directly in the terminal rather than from a startup file. After receiving the first IMU message, the node will prompt you to hold the IMU in a specific orientation and press Enter to record the measurement. After completing all 6 directions, node will calculate the calibration parameters and write them to the specified YAML file. The basic algorithm is based on and similar to the least squares calibration method described in the STMicroelectronics application note. Due to the nature of the algorithm, obtaining good calibration requires the IMU to be positioned fairly precisely along each of its axes.

6.1.1. Calibration steps

Note: During calibration, make sure the robot is still.

jetson motherboard/Raspberry Pi 4B

On the robot side (take jetson nano as an example). Open the terminal and enter the startup command,

```
roslaunch transbot_bringup calibrate_imu.launch
```

Raspberry Pi 5

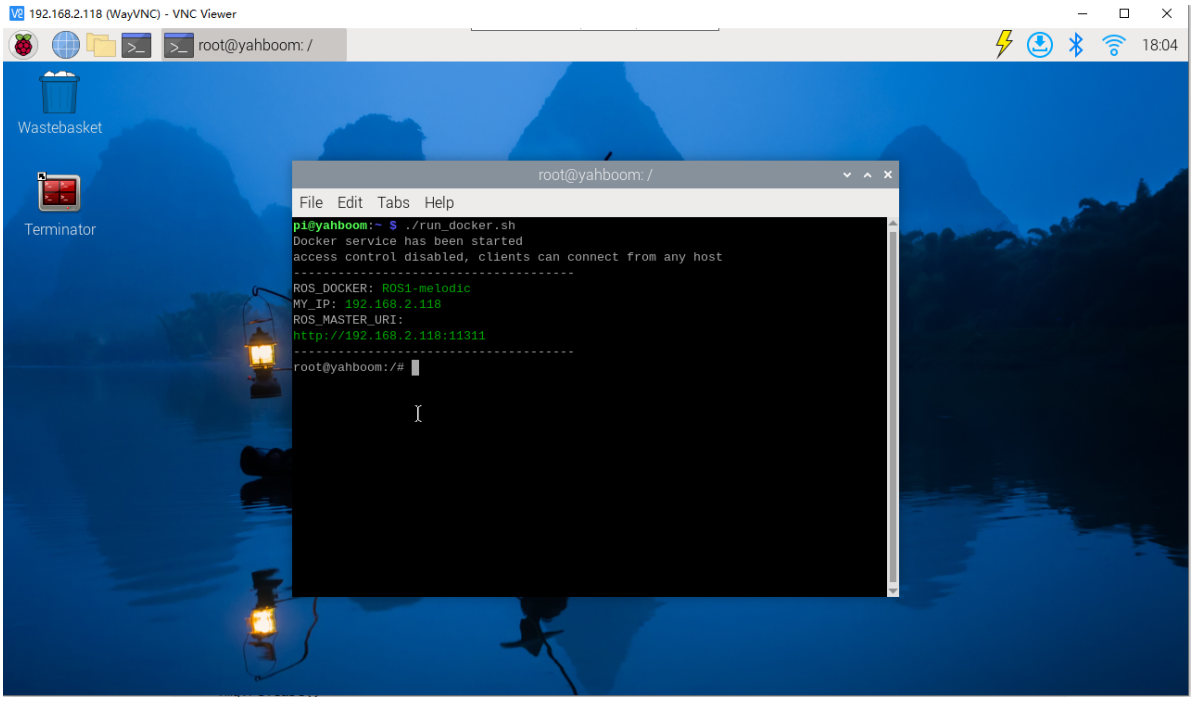
Before running, please confirm that the large program has been permanently closed

Enter docker

Note: If there is a terminal that automatically starts docker, or there is a docker terminal that has been opened, you can directly enter the docker terminal to run the command, and there is no need to manually start docker

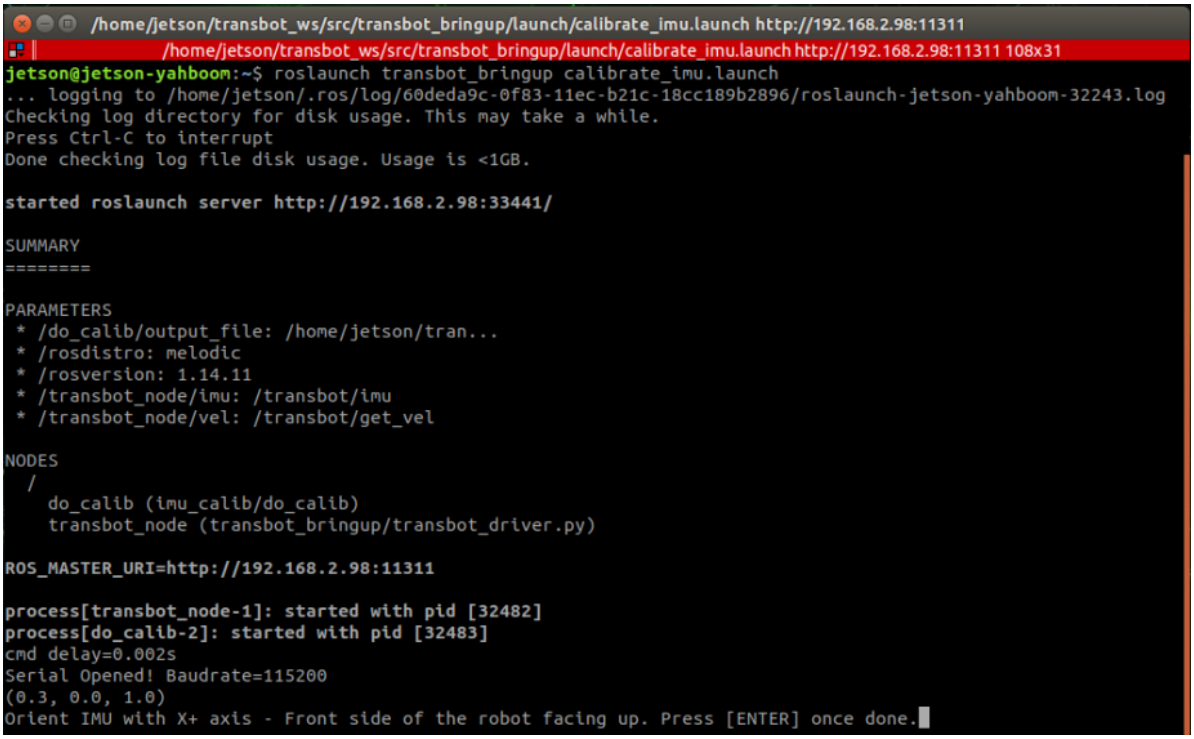
Start docker manually

```
./run_docker.sh
```

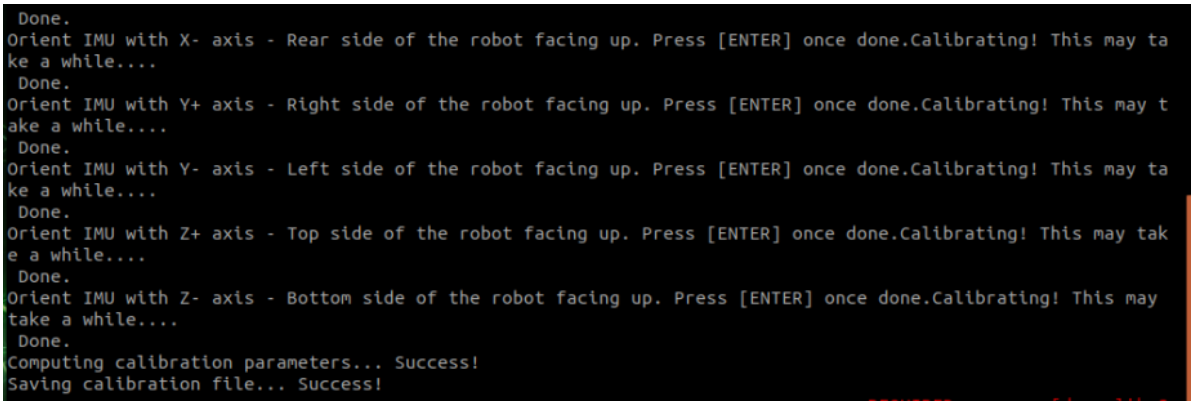


On the robot side (take jetson nano as an example). Open the terminal and enter the startup command,

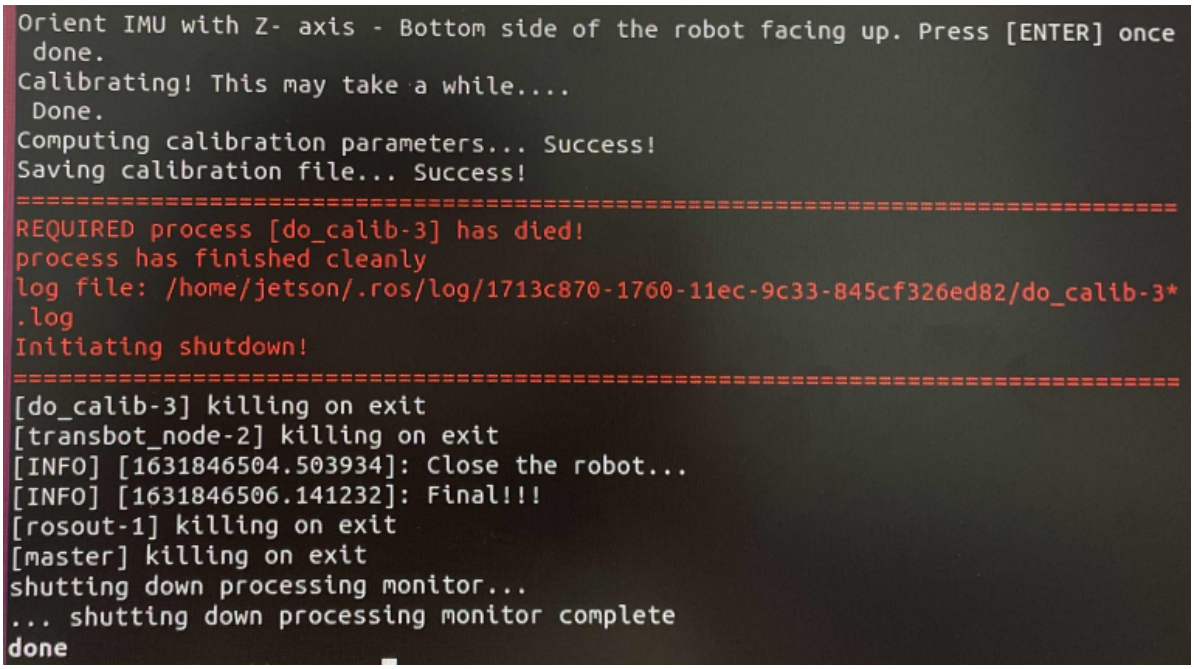
```
roslaunch transbot_bringup calibrate_imu.launch
```



At this time, the calibration begins, and the six axes of the imu [X+, X-, Y+, Y-, Z+, Z-] are calibrated in sequence. At this point, click the [Enter] key repeatedly until the



The calibration is successful and the calibration information is saved in the YAML file. After calibration is completed, it will exit automatically, as shown below:



View node graph

jetson motherboard/Raspberry Pi 4B

```
rqt_graph
```

Raspberry Pi 5

Enter the same docker from multiple terminals

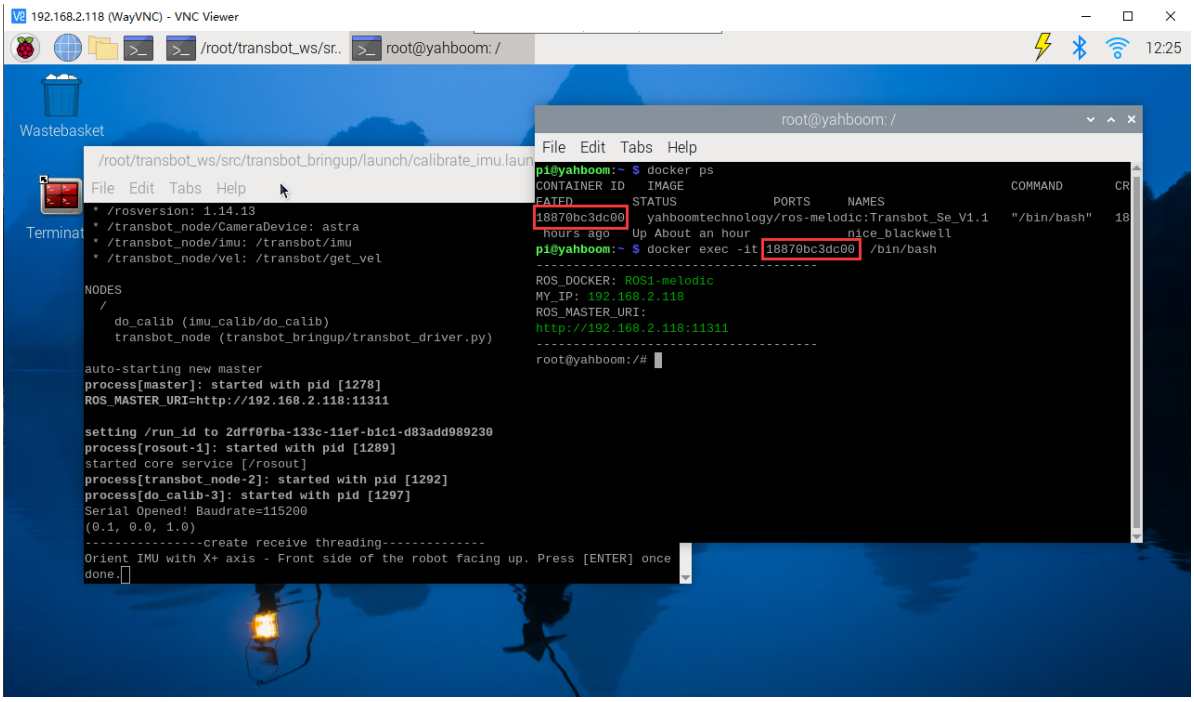
Keep the program of the previous docker terminal running and open a new terminal

Enter the following command

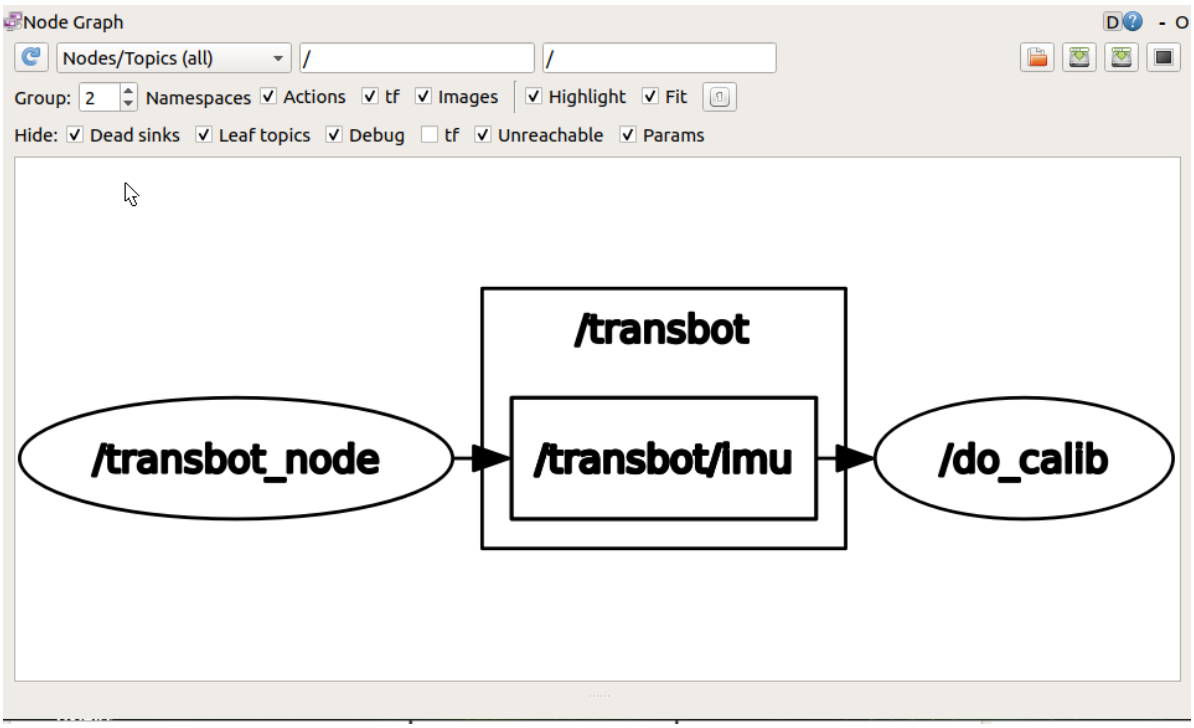
```
docker ps
```

Enter the same docker and use the following 18870bc3dc00 to modify the ID displayed on the actual terminal.

```
docker exec -it 18870bc3dc00 /bin/bash
```




```
rqt_graph
```



6.1.2. Related nodes

transbot_node

Topic	Type	Parsing
/transbot/imu	sensor_msgs/Imu	The most primitive imu data of the car

do_calib node

/	Type	Defaults value	Parsing
Subscribed	sensor_msgs/Imu	/raw_imu	Original, uncalibrated IMU measurement
~calib_file	string	"imu_calib.yaml"	The file where the calibration parameters will be written
~measurements	int	500	Number of measurements collected for each direction
~reference_acceleration	double	9.80665	Acceleration of gravity

6.2. Line speed calibration

Prepare:

- Use a meter ruler to measure the distance of 1 meter and mark it.
- Place the cart at the starting point.
- Modify the parameter [linear_scale] in bringup.launch to 1.0.

jetson motherboard/Raspberry Pi 4B

Start the robot driver (robot side)

```
roslaunch transbot_bringup bringup.launch
```

Line speed calibration (robot side or virtual machine side)

```
roslaunch transbot_bringup calibrate_linear.launch
```

Raspberry Pi 5

Before running, please press Ctrl+c to close all running docker programs.

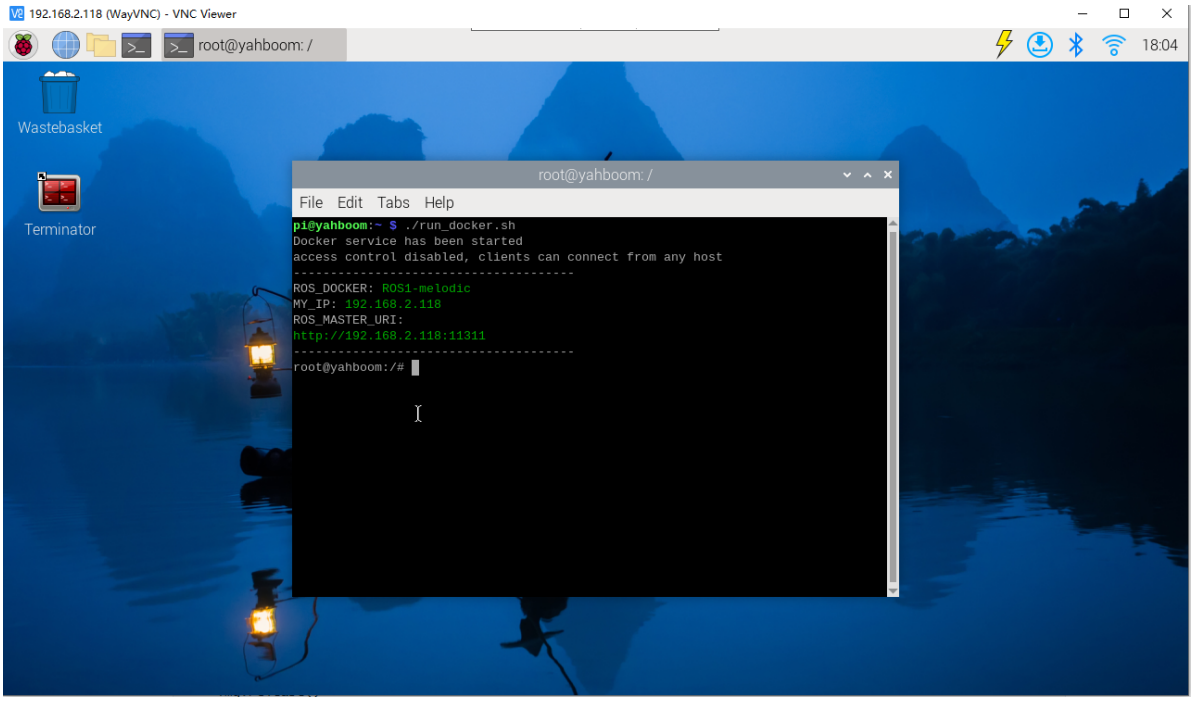
Before running, please confirm that the large program has been permanently closed

Enter docker

Note: If there is a terminal that automatically starts docker, or there is a docker terminal that has been opened, you can directly enter the docker terminal to run the command, and there is no need to manually start docker

Start docker manually

```
./run_docker.sh
```



Start the robot driver (robot side)

```
roslaunch transbot_bringup bringup.launch
```

Enter the same docker from multiple terminals

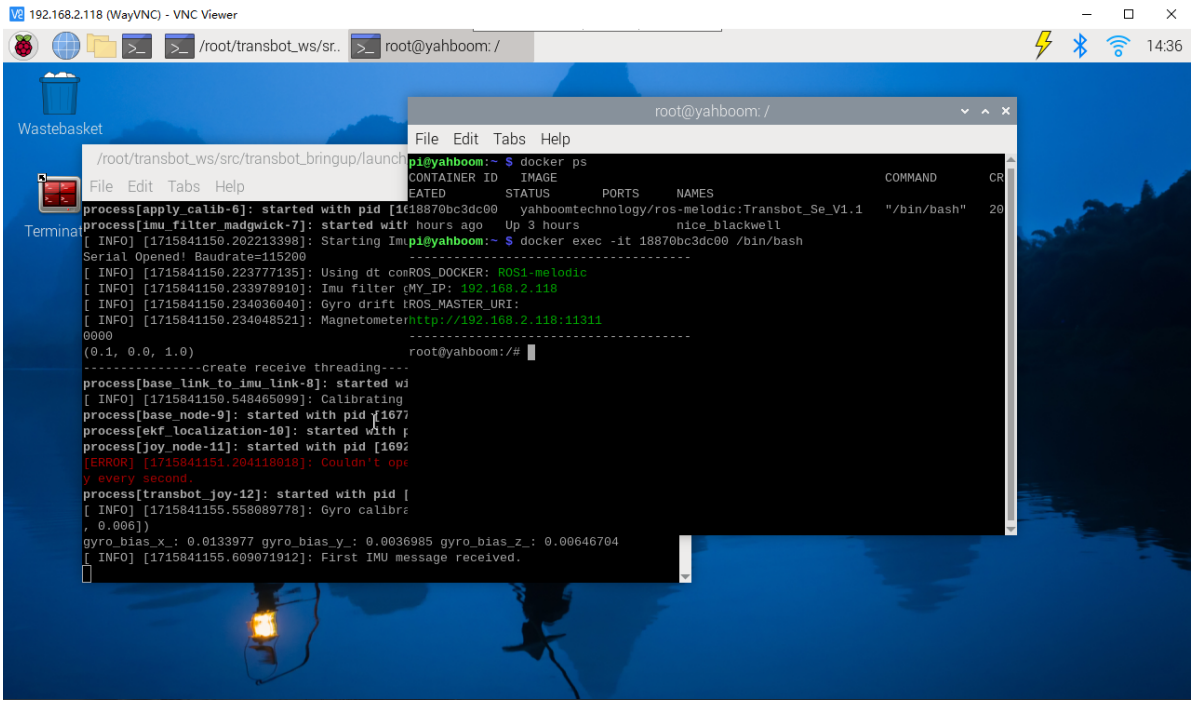
Keep the program of the previous docker terminal running and open a new terminal

Enter the following command

```
docker ps
```

Enter the same docker and use the following 18870bc3dc00 to modify the ID displayed on the actual terminal.

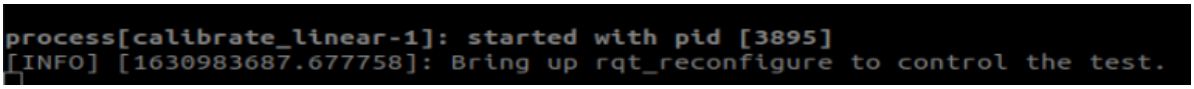
```
docker exec -it 18870bc3dc00 /bin/bash
```



Line speed calibration (robot side or virtual machine side)

```
roslaunch transbot_bringup calibrate_linear.launch
```

The following picture appears:



Enable dynamic parameter adjustment (robot side or virtual machine side)

jetson motherboard/Raspberry Pi 4B

```
rosrun rqt_reconfigure rqt_reconfigure
```

Raspberry Pi 5

Enter the same docker from multiple terminals

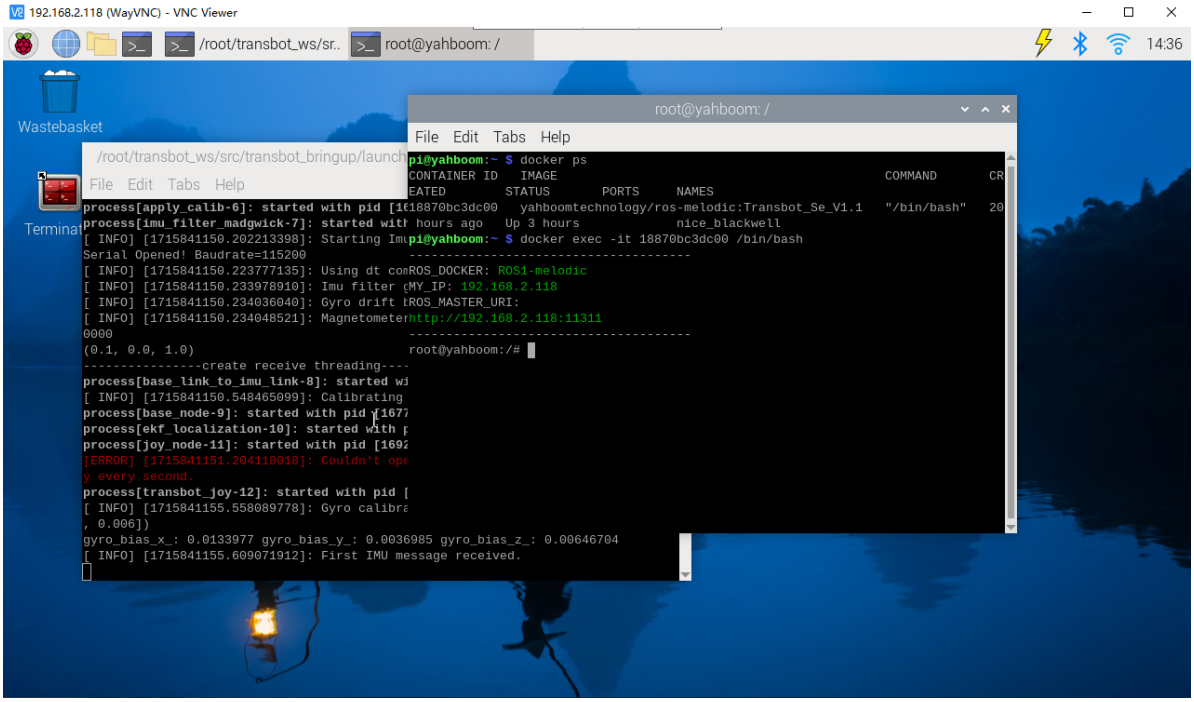
Keep the program of the previous docker terminal running and open a new terminal

Enter the following command

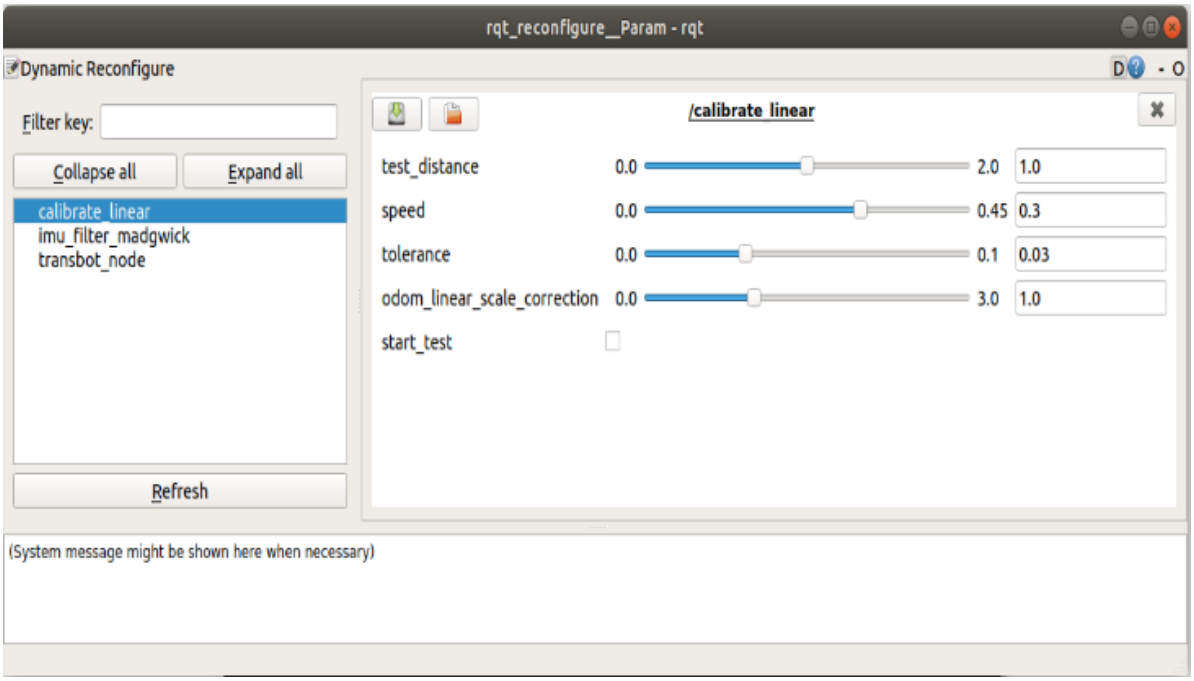
```
docker ps
```

Enter the same docker and use the following 18870bc3dc00 to modify the ID displayed on the actual terminal.

```
docker exec -it 18870bc3dc00 /bin/bash
```




```
rosrun rqt_reconfigure rqt_reconfigure
```



Click the square on the right side of [start_test] and start moving the [test_distance] distance. At this time, observe whether the car actually moves [test_distance]. If not, adjust the parameter [odom_linear_scale_correction], return the car to the starting point and continue testing.

- test_distance: test distance. Not too big, the default is one meter.
- speed: test line speed. If the speed is too great, the inertia will be great.
- tolerance: error in reaching the target. If the error is too small, it will jitter at the target position. On the contrary, the error when reaching the target point will be large.
- odom_linear_scale_correction: Odometer scaling.
- start_test: Start testing.

After the test is completed, remember the value of [odom_linear_scale_correction] and modify it to the value of the parameter [linear_scale] in bringup.launch.

6.3. Angular velocity calibration

Prepare:

- Place the car in a position where it can easily rotate at different angles.
- Modify the parameter [angular_scale] in bringup.launch to 1.0.

jetson motherboard/Raspberry Pi 4B

Start the robot driver (robot side)

```
roslaunch transbot_bringup bringup.launch
```

Angular velocity calibration (robot side or virtual machine side)

```
  roslaunch transbot_bringup calibrate_angular.launch
```

Raspberry Pi 5

Before running, please press Ctrl+c to close all running docker programs.

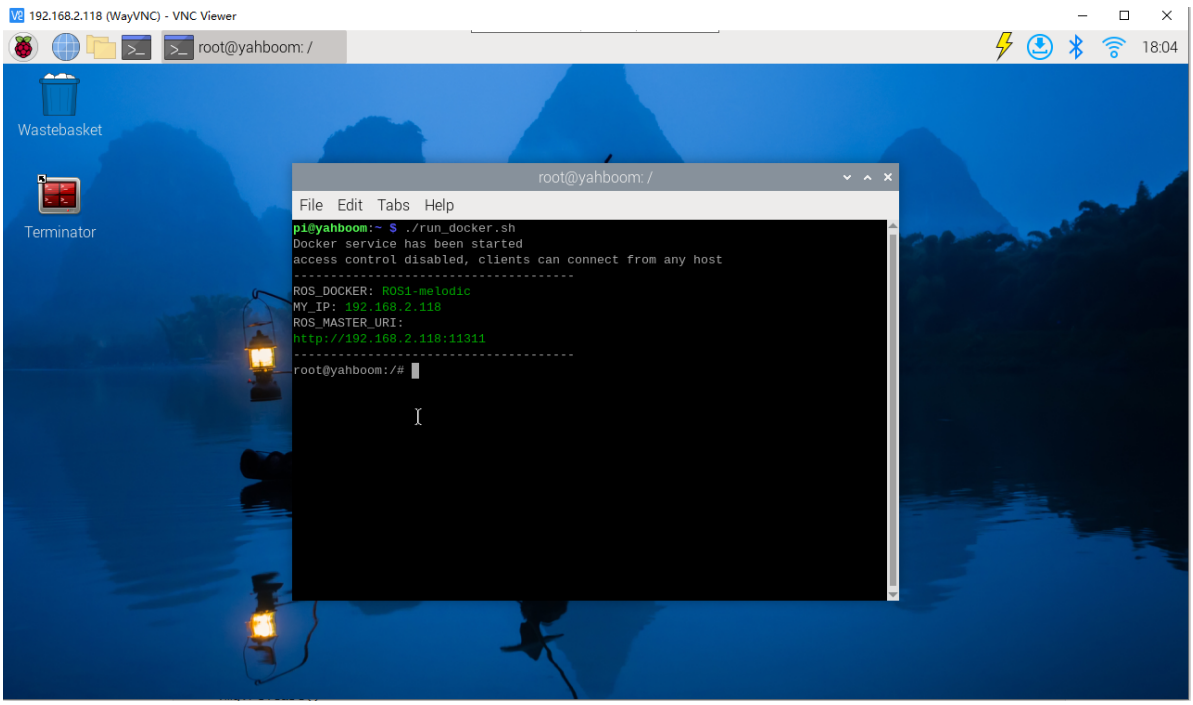
Before running, please confirm that the large program has been permanently closed

Enter docker

Note: If there is a terminal that automatically starts docker, or there is a docker terminal that has been opened, you can directly enter the docker terminal to run the command, and there is no need to manually start docker

Start docker manually

```
./run_docker.sh
```



Start the robot driver (robot side)

```
roslaunch transbot_bringup bringup.launch
```

Enter the same docker from multiple terminals

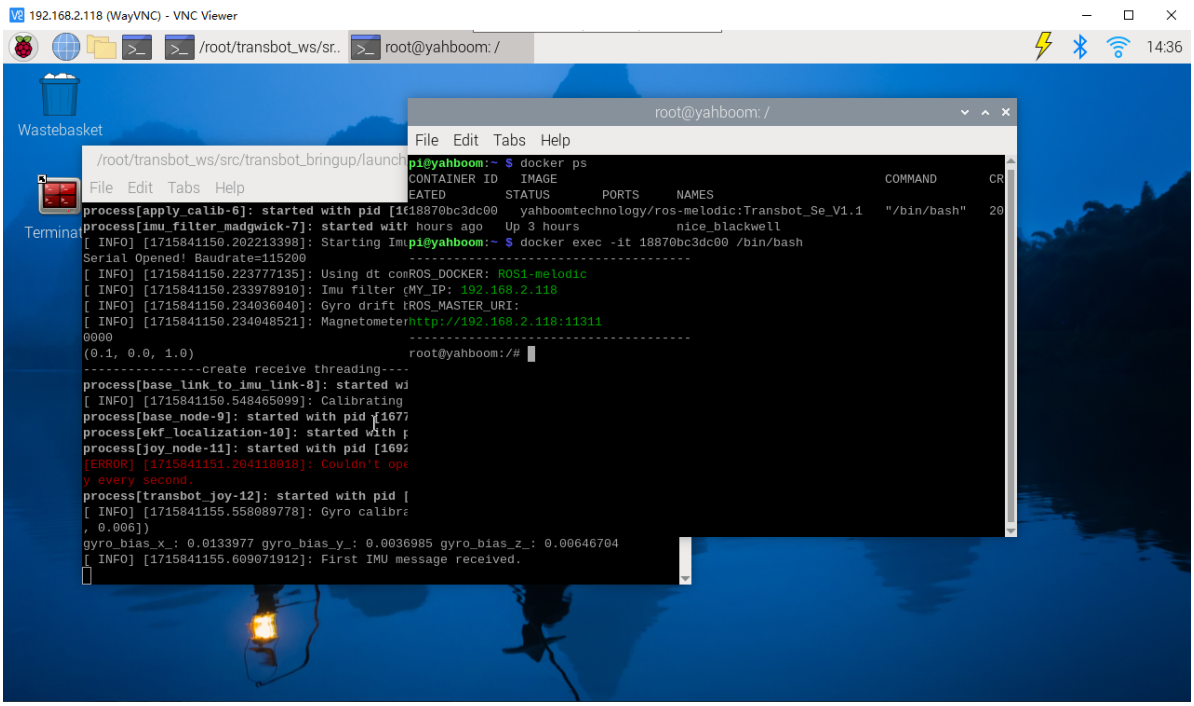
Keep the program of the previous docker terminal running and open a new terminal

Enter the following command

```
docker ps
```

Enter the same docker and use the following 18870bc3dc00 to modify the ID displayed on the actual terminal.

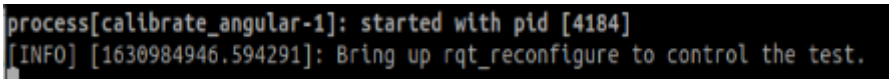
```
docker exec -it 18870bc3dc00 /bin/bash
```



Angular velocity calibration (robot side or virtual machine side)

```
roslaunch transbot_bringup calibrate_angular.launch
```

The following picture appears:



Enable dynamic parameter adjustment (robot side or virtual machine side)

jetson motherboard/Raspberry Pi 4B

```
rosrun rqt_reconfigure rqt_reconfigure
```

Raspberry Pi 5

Enter the same docker from multiple terminals

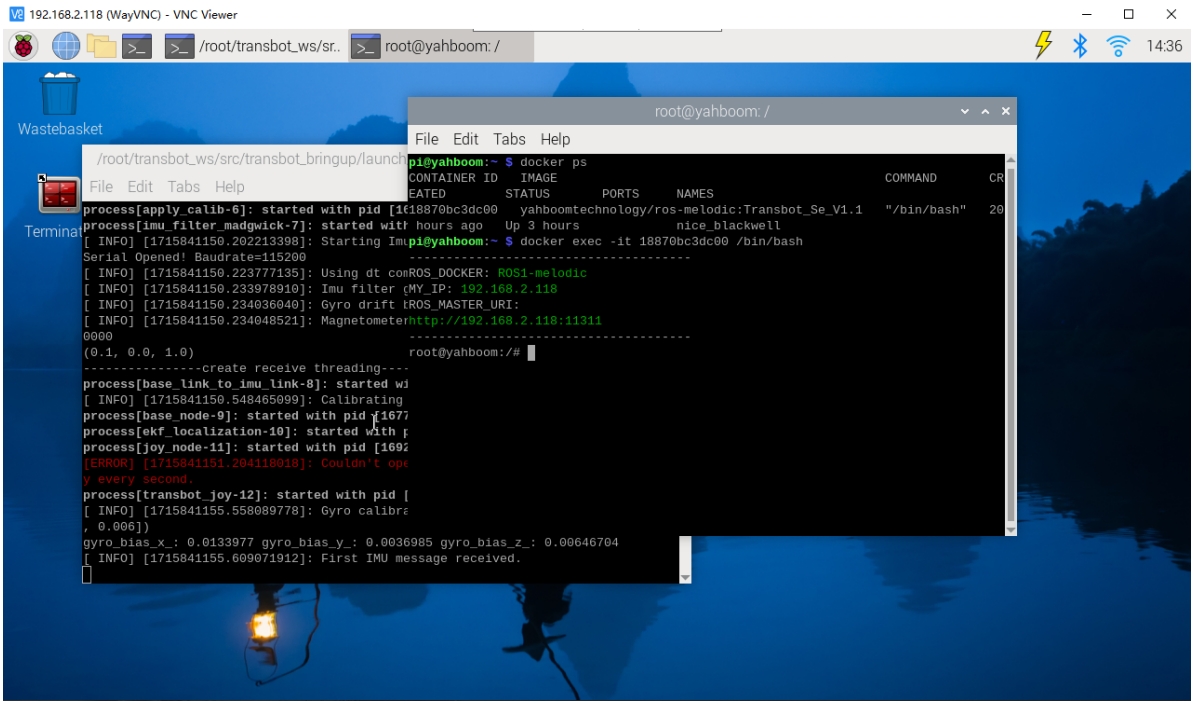
Keep the program of the previous docker terminal running and open a new terminal

Enter the following command

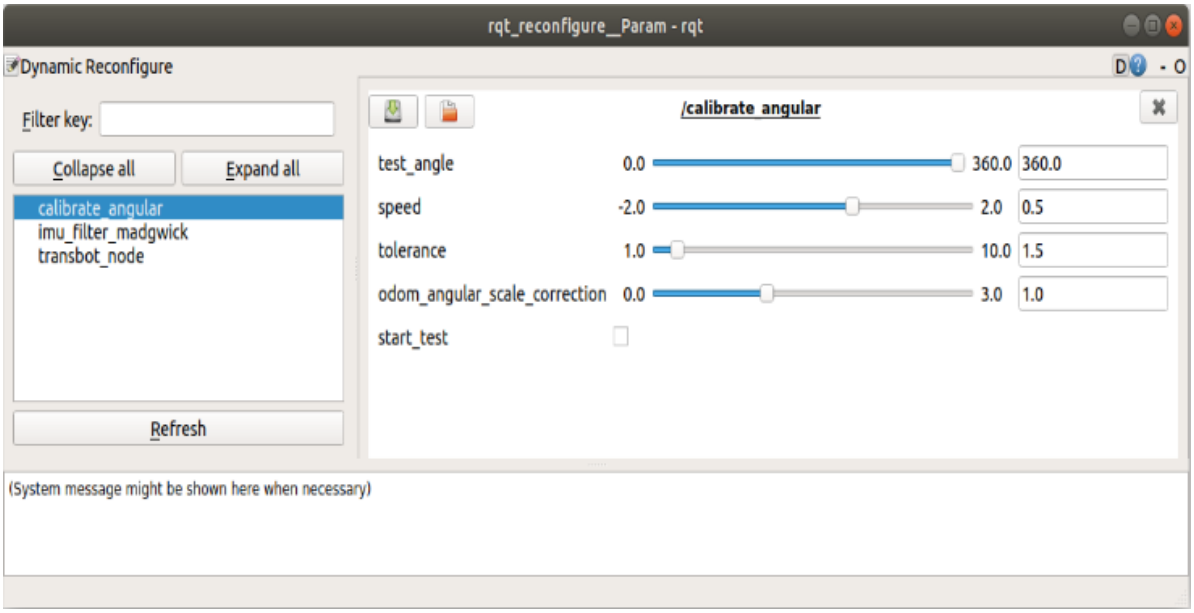
```
docker ps
```

Enter the same docker and use the following 18870bc3dc00 to modify the ID displayed on the actual terminal.

```
docker exec -it 18870bc3dc00 /bin/bash
```




```
rosrun rqt_reconfigure rqt_reconfigure
```



Click the square on the right side of [start_test] and start moving the [test_angle] distance. At this time, observe whether the car actually moves [test_angle]. If not, adjust the parameter [odom_angule_scale_correction], return the car to the starting point and continue testing.

- test_angle: test distance. Not too big, the default is 360°.
- speed: test angular speed. If the speed is too great, the inertia will be great.
- tolerance: error in reaching the target. If the error is too small, it will jitter at the target position. On the contrary, the error when reaching the target point will be large.

- odom_angule_scale_correction: Odometer scaling.
- start_test: Start testing.

After the test is completed, remember the value of [odom_angule_scale_correction] and modify it to the value of the parameter [angular_scale] in bringup.launch.