

Line speed calibration

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.

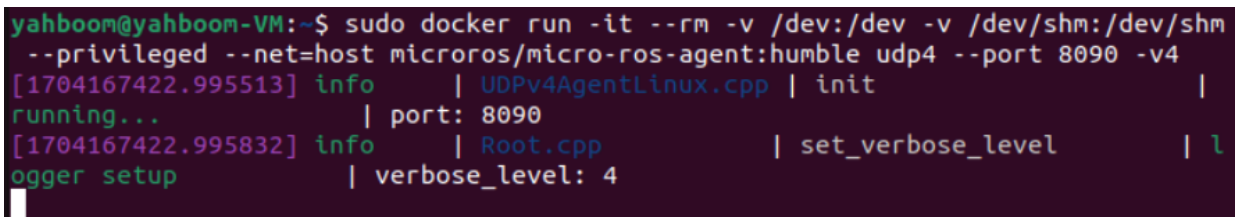
1、 Program function description

The car connects to the agent, runs the program, and adjusts the parameters here through the dynamic parameter regulator to calibrate the linear speed of the car. The intuitive reflection of the calibrated linear speed is to give the car an instruction to go straight forward for 1 meter to see how far it actually ran and whether it is within the error range.

2、 Start and connect to the agent

Taking the supporting 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 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 | 1
ogger 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.

```

[1702630014.015846] Info | ProxyClient.cpp | create_participant | participant created | client_key: 0x0B62A009, participant_id: 0x000(1)
[1702630014.135363] Info | ProxyClient.cpp | create_topic | topic created | client_key: 0x0B62A009, topic_id: 0x000(2), participant_id: 0x000(1)
[1702630014.223689] Info | ProxyClient.cpp | create_publisher | publisher created | client_key: 0x0B62A009, publisher_id: 0x000(3), participant_id: 0x000(1)
[1702630014.415510] Info | ProxyClient.cpp | create_datawriter | datawriter created | client_key: 0x0B62A009, datawriter_id: 0x000(5), publisher_id: 0x000(3)
[1702630014.428530] Info | ProxyClient.cpp | create_topic | topic created | client_key: 0x0B62A009, topic_id: 0x001(2), participant_id: 0x000(1)
[1702630014.527190] Info | ProxyClient.cpp | create_publisher | publisher created | client_key: 0x0B62A009, publisher_id: 0x001(3), participant_id: 0x000(1)
[1702630014.543889] Info | ProxyClient.cpp | create_datawriter | datawriter created | client_key: 0x0B62A009, datawriter_id: 0x001(5), publisher_id: 0x001(3)
[1702630014.554490] Info | ProxyClient.cpp | create_topic | topic created | client_key: 0x0B62A009, topic_id: 0x002(2), participant_id: 0x000(1)
[1702630014.737059] Info | ProxyClient.cpp | create_publisher | publisher created | client_key: 0x0B62A009, publisher_id: 0x002(3), participant_id: 0x000(1)
[1702630014.755072] Info | ProxyClient.cpp | create_datawriter | datawriter created | client_key: 0x0B62A009, datawriter_id: 0x002(5), publisher_id: 0x002(3)
[1702630014.818985] Info | ProxyClient.cpp | create_topic | topic created | client_key: 0x0B62A009, topic_id: 0x003(2), participant_id: 0x000(1)
[1702630014.840001] Info | ProxyClient.cpp | create_subscriber | subscriber created | client_key: 0x0B62A009, subscriber_id: 0x000(4), participant_id: 0x000(1)
[1702630014.864010] Info | ProxyClient.cpp | create_datareader | datareader created | client_key: 0x0B62A009, datareader_id: 0x000(6), subscriber_id: 0x000(4)
[1702630014.959908] Info | ProxyClient.cpp | create_topic | topic created | client_key: 0x0B62A009, topic_id: 0x004(2), participant_id: 0x000(1)
[1702630015.033537] Info | ProxyClient.cpp | create_subscriber | subscriber created | client_key: 0x0B62A009, subscriber_id: 0x001(4), participant_id: 0x000(1)
[1702630015.140350] Info | ProxyClient.cpp | create_datareader | datareader created | client_key: 0x0B62A009, datareader_id: 0x001(6), subscriber_id: 0x001(4)
[1702630015.158510] Info | ProxyClient.cpp | create_topic | topic created | client_key: 0x0B62A009, topic_id: 0x005(2), participant_id: 0x000(1)
[1702630015.241039] Info | ProxyClient.cpp | create_subscriber | subscriber created | client_key: 0x0B62A009, subscriber_id: 0x002(4), participant_id: 0x000(1)
[1702630015.347393] Info | ProxyClient.cpp | create_datareader | datareader created | client_key: 0x0B62A009, datareader_id: 0x002(6), subscriber_id: 0x002(4)

```

3、starting program

First, start the car's underlying data processing program. This program will release the TF transformation of odom->base_footprint. With this TF change, you can calculate "how far the car has gone" and input it at the terminal.

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

```

[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 arguments
[static_transform_publisher-6] [WARN] [1702865272.984740987] []: Old-style arguments are deprecated; see --help for new-style arguments
[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 the robot_description topic...

```

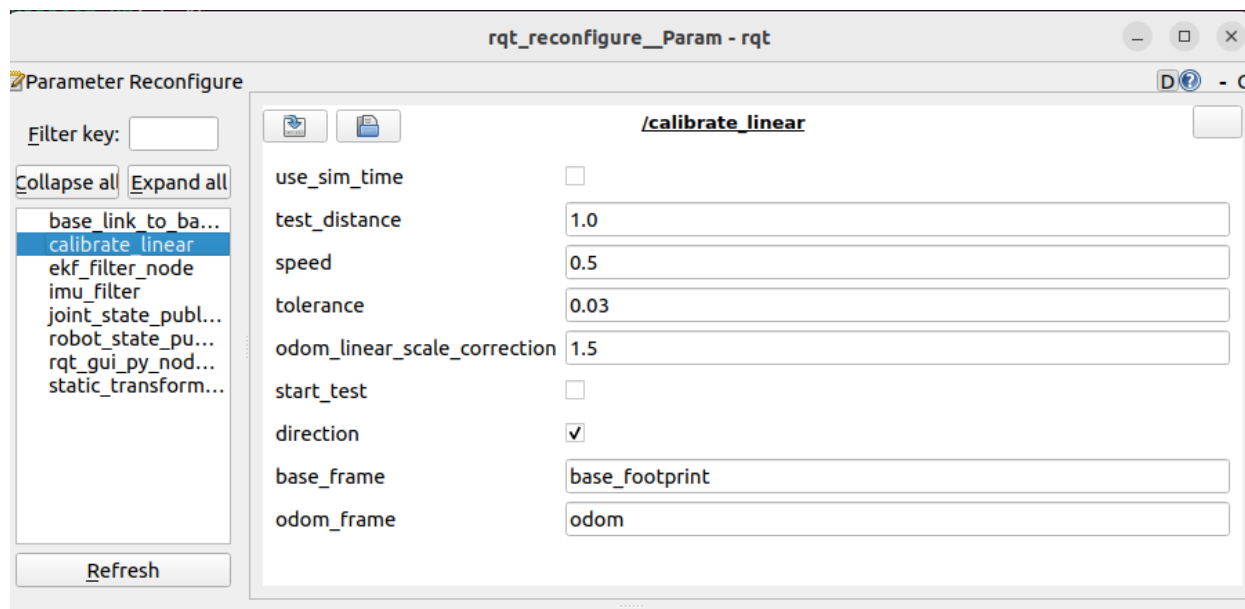
Then, start the car linear speed calibration program and enter in the terminal,

```
ros2 run yahboomcar_bringup calibrate_linear
```

```
yahboom@yahboom-VM:~$ ros2 run yahboomcar_bringup calibrate_linear
finish init work
Please change the state!
Please change the state!
Please change the state!
Please change the state!
Please change the state!
Please change the state!
Please change the state!
Please change the state!
Please change the state!
Please change the state!
Please change the state!
Please change the state!
Please change the state!
Please change the state!
Please change the state!
Please change the state!
Please change the state!
Please change the state!
```

Finally, open the dynamic parameter adjuster and enter in the terminal,

```
ros2 run rqt_reconfigure rqt_reconfigure
```



(System message might be shown here when necessary)

Note: There may not be the above nodes when you first open it. You can see all nodes after clicking Refresh. The displayed **calibrate_linear** node is the node for calibrating linear speed.

4、 Start calibration

In the rqt_reconfigure interface, select the calibrate_linear node. There is **start_test** below and click the box to the right to start calibration. Other parameters in the rqt interface are explained as follows:

- test_distance: Calibrate the test distance, here the test goes forward 1 meter;
- speed: Linear speed;
- tolerance: tolerance for error;
- odom_linear_scale_correction: Linear speed proportional coefficient, if the test result is not ideal, just modify this value;
- start_test: test switch:
- direction: It can be ignored. This value is used for the wheel structure trolley. After modification, the linear speed of left and right movement can be calibrated;
- base_frame: The name of the base coordinate system;
- odom_frame: The name of the odometer coordinate system.

Click start_test to start calibration. The car will monitor the TF transformation of base_footprint and odom, calculate the theoretical distance traveled by the car, wait until the error is less than the tolerance, and issue a parking instruction.

```
distance: 0.0
error: -1.0
distance: 0.6076996120869244
error: -0.39230038791307564
distance: 0.6270069712037827
error: -0.3729930287962173
distance: 0.7154783516427078
error: -0.2845216483572922
distance: 0.7632770836095988
error: -0.23672291639040122
distance: 0.7735690887805375
error: -0.22643091121946246
distance: 0.7735690887805375
error: -0.22643091121946246
distance: 0.8526809501167965
error: -0.14731904988320355
distance: 0.8727603064439674
error: -0.1272396935560326
distance: 0.9577208605992955
error: -0.04227913940070449
distance: 0.9577208605992955
error: -0.04227913940070449
distance: 1.0235067225409535
error: 0.02350672254095354
done
```

If the actual distance the car runs is not 1m, then modify the odom_linear_scale_correction parameter in rqt. After modification, click on the blank space, click start_test again, reset start_test, and then click start_test again to calibrate. The same goes for modifying other parameters. You need to click on the blank space to write the modified parameters.

5、 Code analysis

Source code reference path (taking the supporting virtual machine as an example):

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

calibrate_linear.py, the core code is as follows,

```

#Monitor the TF transformation of base_footprint and odom
def get_position(self):
    try:
        now = rclpy.time.Time()
        trans = self.tf_buffer.lookup_transform(self.odom_frame, self.base_frame, now)
        #print("trans: ",trans)
        return trans
    except (LookupException, ConnectivityException, ExtrapolationException):
        self.get_logger().info('transform not ready')
        raise
    return

#Get the current coordinates
self.position.x = self.get_position().transform.translation.x
self.position.y = self.get_position().transform.translation.y
#Calculate how far distance and error are
distance = sqrt(pow((self.position.x - self.x_start), 2) + pow((self.position.y -
self.y_start), 2))
distance *= self.odom_linear_scale_correction
print("distance: ",distance)
error = distance - self.test_distance

```