2. Robot PID debugging

Note: The PID has been debugged before the product leaves the factory. It is not recommended to adjust it yourself, as it may cause problems with subsequent functions. But you can learn

Learn debugging methods and operating procedures.

According to different models, just set the purchased model in [.bashrc], X1 (normal four-wheel drive) X3 (Mailun) X3plus (Mailun robotic arm) R2 (Ackerman differential) etc. , this section takes X3 as an example:

```
sudo vim .bashrc
```

Find the [ROBOT_TYPE] parameter and modify the corresponding model

```
export ROBOT_TYPE=X3 # ROBOT_TYPE: X1 X3 X3plus R2 X7
```

2.1. Start the chassis

```
#Raspberry Pi 5 master needs to enter docker first, please perform this step
#If running the script into docker fails, please refer to ROS/07, Docker tutorial
~/run_docker.sh
```

2.1.1. Function package path

```
~/yahboom_ws/src/yahboom_bringup/
```

2.1.2. Start

```
roslaunch yahboomcar_bringup bringup.launch #robotchassisstart
rosrun rqt_reconfigure rqt_reconfigure # Parameter adjuster
rqt_plot # rqt visualization tool
rosrun yahboomcar_ctrl yahboom_keyboard.py #Keyboard control node
```

1), first check the source code Mcnamu_driver.py

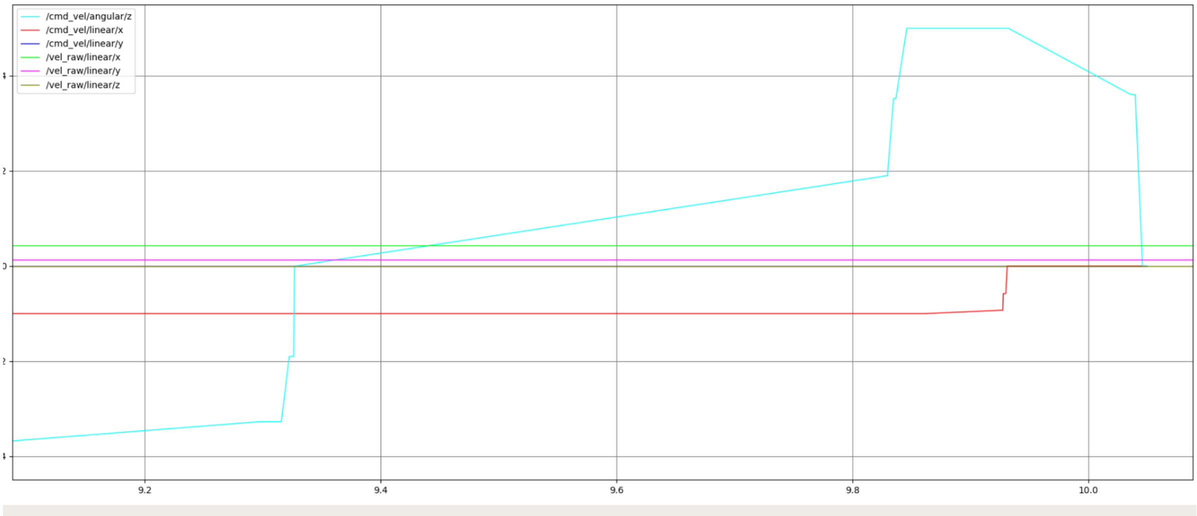
```
def dynamic_reconfigure_callback(self, config, level):
         # self.car.set_pid_param(config['Kp'], config['Ki'], config['Kd'])
         # print("PID: ", config['Kp'], config['Ki'], config['Kd'])
         self.linear_max = config['linear_max']
         self.linear_min = config['linear_min']
         self.angular_max = config['angular_max']
         self.angular_min = config['angular_min']
         return config
```

Find the line self.car.set_pid_param and remove the comment # in front of it to debug the robot's PID.

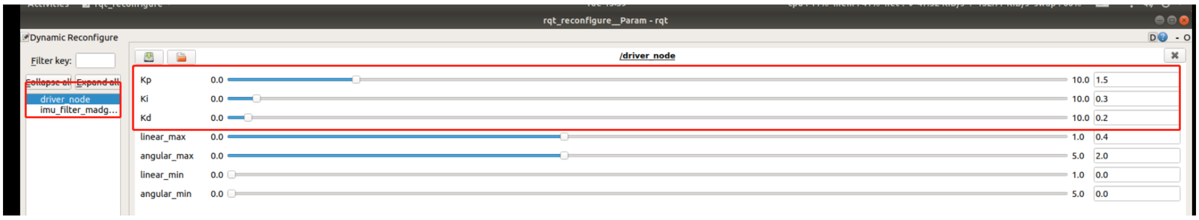
2. First select the debugging parameters. After entering the name of each parameter as shown in the figure below, click [+] to add, [-] to delete.

- /cmd_vel/angular/z: target angular velocity
- /cmd_vel/linear/x: Linear velocity in the target x direction
- /cmd_vel/linear/y: Linear velocity in the target y direction
- /vel_raw/angular/z: Current angular velocity
  -/vel_raw/linear/x: current linear velocity in the x direction
  -/vel_raw/linear/y: current linear velocity in the y direction

Note: The speed in the y direction is only available for the Mecanum wheel car X1



3. In the rqt_reconfigure window, as shown in the figure, select the [driver_node] node. We only need to adjust the three parameters [Kp], [Ki], and [Kd], and do not adjust the others.



Kp	0.0	10.0	1.5
Ki	0.0	10.0	0.3
Kd	0.0	10.0	0.2
linear_max	0.0	1.0	0.4
angular_max	0.0	5.0	2.0
linear_min	0.0	1.0	0.0
angular_min	0.0	5.0	0.0

Debugging steps:

- Use the keyboard keys on the keyboard control terminal to drive the car to move [forward], [backward], [turn left], and [turn right]
- Observe the changes in the [rqt_plot] window image so that the current angular velocity is close to the target angular velocity, and the current linear velocity is close to the target linear velocity. It is impossible to overlap. The target speed is in an ideal state without any interference, and the speed can be increased instantly.
- Use the keyboard [q], [z], [w], [x], [e], [c] to increase or decrease the speed and test the status of different speeds. [t] switches the car linear speed X/Y axis direction, [s] stops keyboard control. **Note that the linear velocity range of the car's XY axis is v_x=[-1.0, 1.0], v_y=[-1.0, 1.0], and the angular velocity range is v_z=[-5.0, 5.0].**
- Observe the changes of [rqt_plot], adjust [Kp], [Ki], and [Kd] data through [rqt_reconfigure], test multiple times, and select the optimal data.

Keyboard control instructions (**Before keyboard control, you need to click with the mouse to open the keyboard control terminal before you can control it**)

- Directional control

[i] or [I]	[linear, 0]	[u] or [U]	[linear, angular]
[,]	[-linear, 0]	[o] or [O]	[linear, - angular]
[j] or [J]	[0, angular]	[m] or [M]	[- linear, - angular]
[l] or [L]	[0, - angular]	[.]	[- linear, angular]

- speed control

Button	Speed change	Button	Speed change
[q]	Linear speed and angular speed are both increased by 10%	[z]	Linear speed and angular speed are both reduced by 10%
[w]	Only the linear speed increases by 10%	[x]	Only the linear speed decreases by 10%
[e]	Only the angular velocity increases by 10%	[c]	Only the angular velocity decreases by 10%
[t]	Line speed X-axis/Y-axis direction switching	[s]	Stop keyboard control

After debugging, the PID is automatically stored in the PCB. Just comment out the PID setting part according to the way you originally viewed the source code.

Note: Other gameplay requires PID debugging, the theory is the same, theoretical reference.