Multi-machine keyboard control

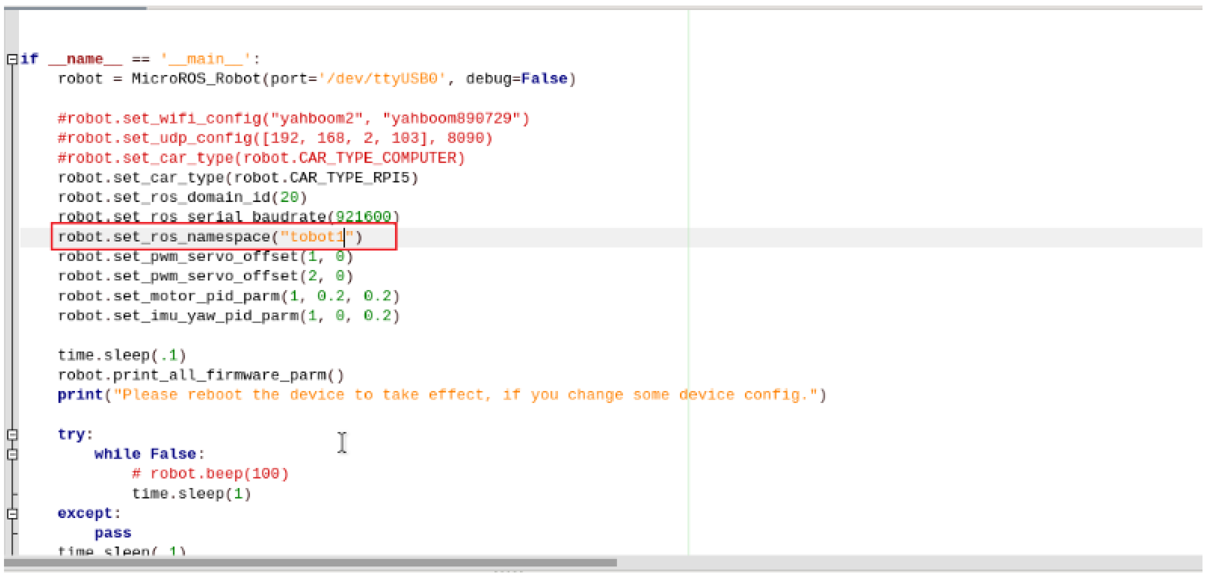
Note: The ROS_DOMAIN_ID of the Raspberry Pi and the microROS control board need to be consistent. You can check **[MicroROS Control Board Parameter Configuration]** to set the microROS control board ROS_DOMAIN_ID. Check the tutorial **[Connect MicroROS Agent]** to determine whether the IDs are consistent.

1、Program function description

After the program is started, the two cars can be controlled to move synchronously through the keyboard.

2、Multi-machine function basic settings

Taking two robots as an example, it is recommended to use two robots equipped with Raspberry Pi and change the **config_robot.py** files respectively. Set robot.set_ros_namespace() to robot1 and robot2 respectively . And **the ROS_DOMAIN_ID of the two cars and the ROS_DOMAIN_ID of the virtual machine need to be set to the same**. Then open the terminal in the /home/pi directory and enter `sudo python3 config_robot.py` to run this program (you need to change it back and re-run this program to run other programs except multi-car).

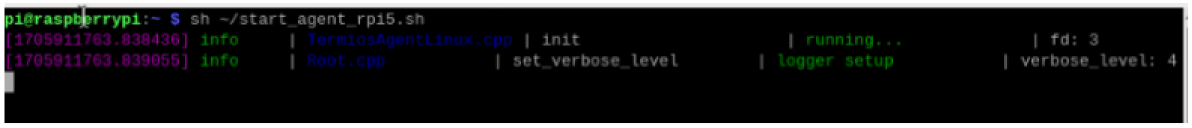


```
if __name__ == '__main__':
    robot = MicroROS_Robot(port='/dev/ttyUSB0', debug=False)

    #robot.set_wifi_config("yahboom2", "yahboom890729")
    #robot.set_udp_config([192, 168, 2, 103], 8090)
    #robot.set_car_type(robot.CAR_TYPE_COMPUTER)
    robot.set_car_type(robot.CAR_TYPE_RPI5)
    robot.set_ros_domain_id(20)
    robot.set_ros_serial_baudrate(921600)
    robot.set_ros_namespace("tobot1")
    robot.set_pwm_servo_offset(1, 0)
    robot.set_pwm_servo_offset(2, 0)
    robot.set_motor_pid_parm(1, 0.2, 0.2)
    robot.set_imu_yaw_pid_parm(1, 0, 0.2)

    time.sleep(.1)
    robot.print_all_firmware_parm()
    print("Please reboot the device to take effect, if you change some device config.")

    try:
        while False:
            # robot.beep(100)
            time.sleep(1)
    except:
        pass
    time.sleep(.1)
```

3、Start and connect to the agent

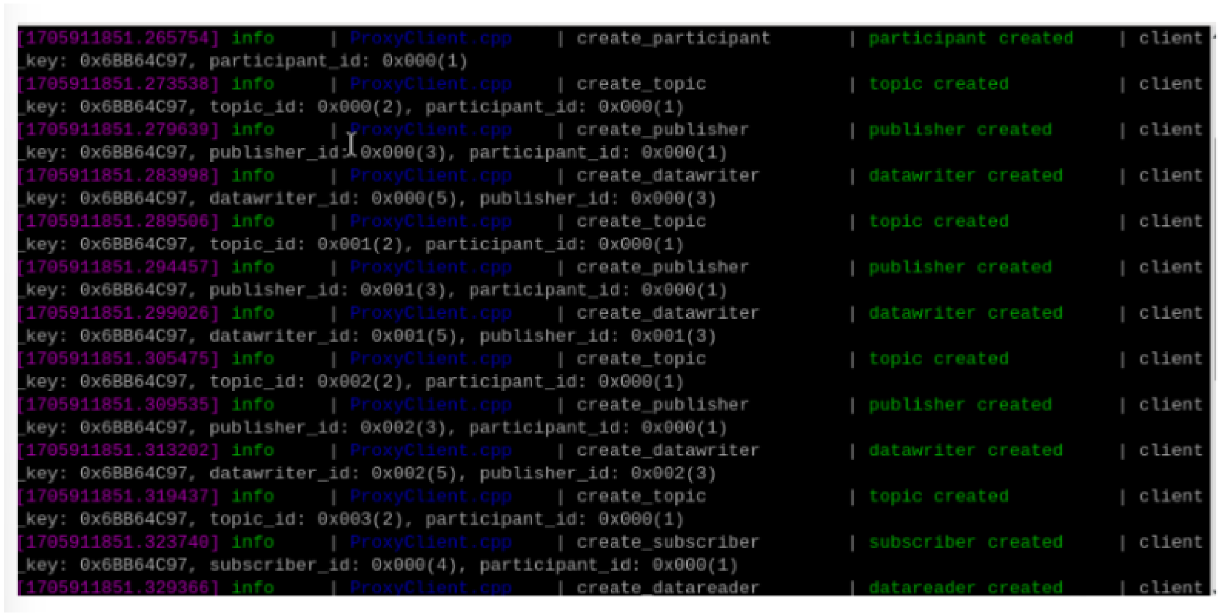
After the Raspberry Pi is successfully powered on, open the terminal and enter the following command to open the agent.

```
sh ~/start_agent_rpi5.sh
```



```
pi@raspberrypi:~ $ sh ~/start_agent_rpi5.sh
[1705911763.838436] info     | TermiosAgentLinux.cpp | init                  | running...             | fd: 3
[1705911763.839055] info     | Root.cpp              | set_verbose_level     | logger setup           | verbose_level: 4
```

Press the reset button on the microROS control board and wait for the car to connect to the agent. The connection is successful as shown in the figure below.



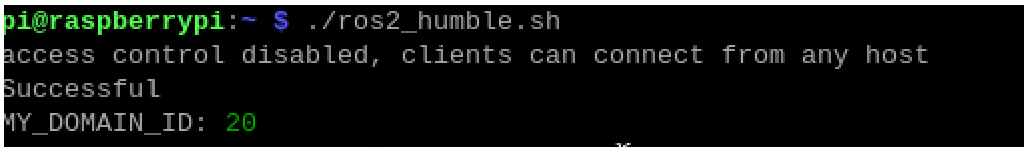
```
[1705911851.265754] info     | ProxyClient.cpp    | create_participant       | participant created     | client
_key: 0x6BB64C97, participant_id: 0x000(1)
[1705911851.273538] info     | ProxyClient.cpp    | create_topic             | topic created           | client
_key: 0x6BB64C97, topic_id: 0x000(2), participant_id: 0x000(1)
[1705911851.279639] info     | ProxyClient.cpp    | create_publisher         | publisher created       | client
_key: 0x6BB64C97, publisher_id: 0x000(3), participant_id: 0x000(1)
[1705911851.283998] info     | ProxyClient.cpp    | create_datawriter        | datawriter created      | client
_key: 0x6BB64C97, datawriter_id: 0x000(5), publisher_id: 0x000(3)
[1705911851.289506] info     | ProxyClient.cpp    | create_topic             | topic created           | client
_key: 0x6BB64C97, topic_id: 0x001(2), participant_id: 0x000(1)
[1705911851.294457] info     | ProxyClient.cpp    | create_publisher         | publisher created       | client
_key: 0x6BB64C97, publisher_id: 0x001(3), participant_id: 0x000(1)
[1705911851.299026] info     | ProxyClient.cpp    | create_datawriter        | datawriter created      | client
_key: 0x6BB64C97, datawriter_id: 0x001(5), publisher_id: 0x001(3)
[1705911851.305475] info     | ProxyClient.cpp    | create_topic             | topic created           | client
_key: 0x6BB64C97, topic_id: 0x002(2), participant_id: 0x000(1)
[1705911851.309535] info     | ProxyClient.cpp    | create_publisher         | publisher created       | client
_key: 0x6BB64C97, publisher_id: 0x002(3), participant_id: 0x000(1)
[1705911851.313202] info     | ProxyClient.cpp    | create_datawriter        | datawriter created      | client
_key: 0x6BB64C97, datawriter_id: 0x002(5), publisher_id: 0x002(3)
[1705911851.319437] info     | ProxyClient.cpp    | create_topic             | topic created           | client
_key: 0x6BB64C97, topic_id: 0x003(2), participant_id: 0x000(1)
[1705911851.323740] info     | ProxyClient.cpp    | create_subscriber        | subscriber created      | client
_key: 0x6BB64C97, subscriber_id: 0x000(4), participant_id: 0x000(1)
[1705911851.329366] info     | ProxyClient.cpp    | create_datareader        | datareader created      | client
```

4、 Enter the car docker

Open another terminal and enter the following command to enter docker.

```
sh ros2_humble.sh
```

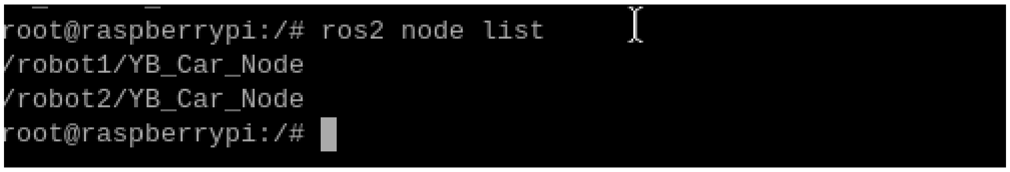
When the following interface appears, you have successfully entered docker. Now you can control the car through commands.



```
pi@raspberrypi:~ $ ./ros2_humble.sh
access control disabled, clients can connect from any host
Successful
MY_DOMAIN_ID: 20
```

Check the currently started node. Choose one of the two Raspberry Pis, open the terminal and enter the following command.

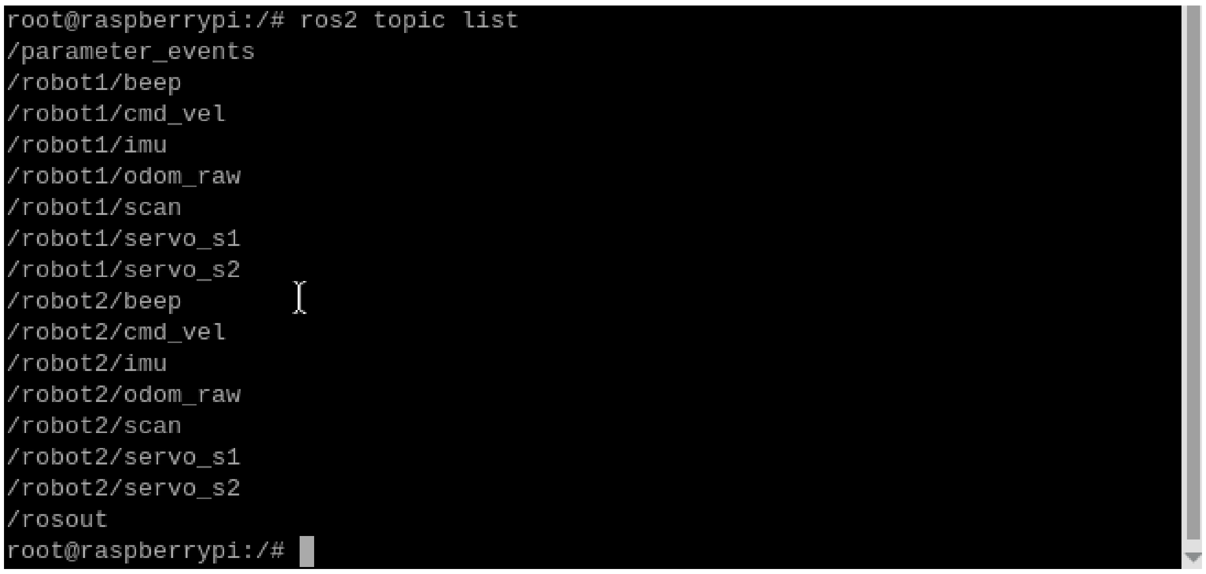
```
ros2 node list
```



```
root@raspberrypi:/# ros2 node list
/robot1/YB_Car_Node
/robot2/YB_Car_Node
root@raspberrypi:/#
```

As shown in the picture above, the nodes of both cars have been started. To query the current topic information, enter in the terminal.

```
ros2 topic list
```

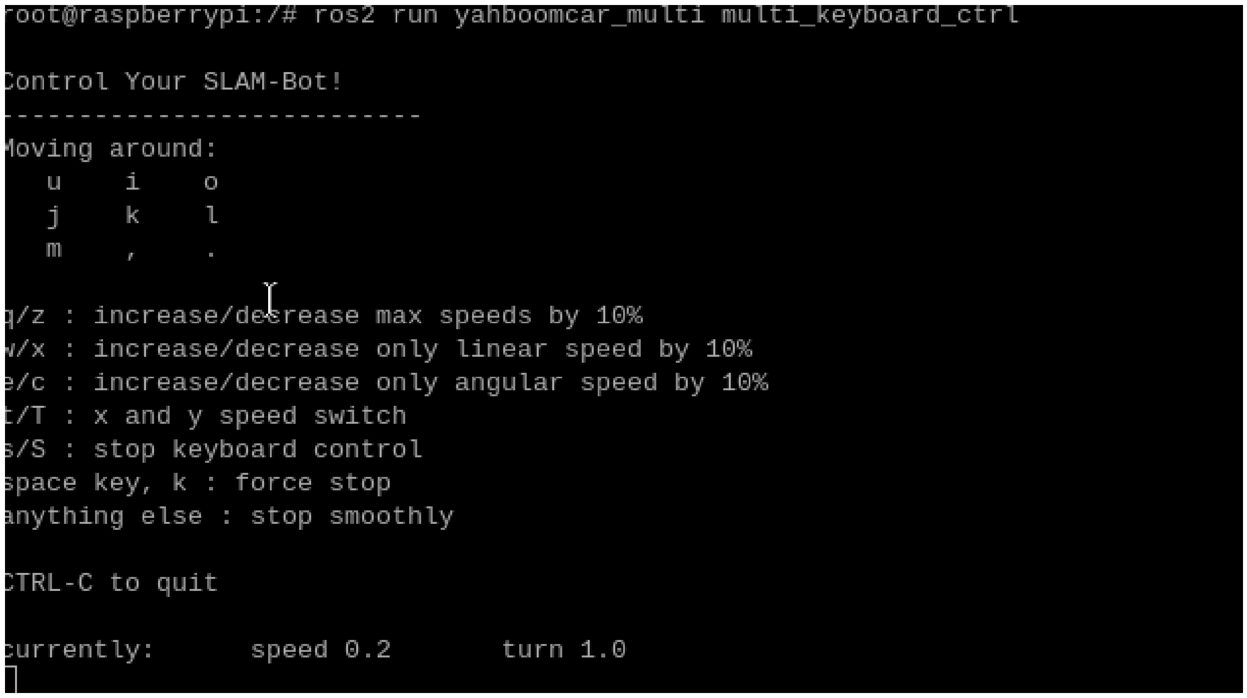


```
root@raspberrypi:/# ros2 topic list
/parameter_events
/robot1/beep
/robot1/cmd_vel
/robot1/imu
/robot1/odom_raw
/robot1/scan
/robot1/servo_s1
/robot1/servo_s2
/robot2/beep
/robot2/cmd_vel
/robot2/imu
/robot2/odom_raw
/robot2/scan
/robot2/servo_s1
/robot2/servo_s2
/rosout
root@raspberrypi:/#
```

5、Start the keyboard control program

Choose one of the two Raspberry Pis, open the terminal and enter the command.

```
ros2 run yahboomcar_multi multi_keyboard_ctrl
```



```
oot@raspberrypi:/# ros2 run yahboomcar_multi multi_keyboard_ctrl

ontrol Your SLAM-Bot!
-------------------------
oving around:
   u    i    o
   j    k    l
   m    ,    .

/z : increase/decrease max speeds by 10%
/x : increase/decrease only linear speed by 10%
/c : increase/decrease only angular speed by 10%
/T : x and y speed switch
/S : stop keyboard control
pace key, k : force stop
nything else : stop smoothly

TRL-C to quit

urrently:      speed 0.2       turn 1.0
```

Keyboard key descriptions are as follows

Directional control:

keyboard keys	Function Description	keyboard keys	Function Description
【i】 or 【I】	【linear，0】	【u】 or 【U】	【linear，angular】
【，】	【-linear，0】	【o】 or 【O】	【linear，-angular】
【j】 or 【J】	【0，linear】	【m】 or 【M】	【-linear，-angular】
【l】 or 【L】	【0，-linear】	【.】	【-linear，angular】

That is to say, install [i] to go forward, press [,] to go back, press [l] to rotate to the right, press [j] to rotate to the left, and so on.

speed control：

keyboard keys	Function Description	keyboard keys	Function Description
【q】	Linear speed and angular speed increased by 10%	【z】	Linear speed and angular speed are reduced by 10%
【w】	Linear speed increased by 10%	【x】	Line speed reduced by 10%
【e】	Angular speed increased by 10%	【c】	Angular speed reduced by 10%
【t】	Linear speed X-axis and Y-axis direction switching	【s】	Stop keyboard control

Note: Since the car has a four-wheel drive structure with ordinary tires and cannot move sideways, the [t] button has no meaning. Before each use of keyboard control, you need to click on the terminal that starts the program, otherwise the key event cannot be detected.

6、Code analysis

Source code reference path:

```
/root/yahboomcar_ws/src/yahboomcar_multi/yahboomcar_multi
```

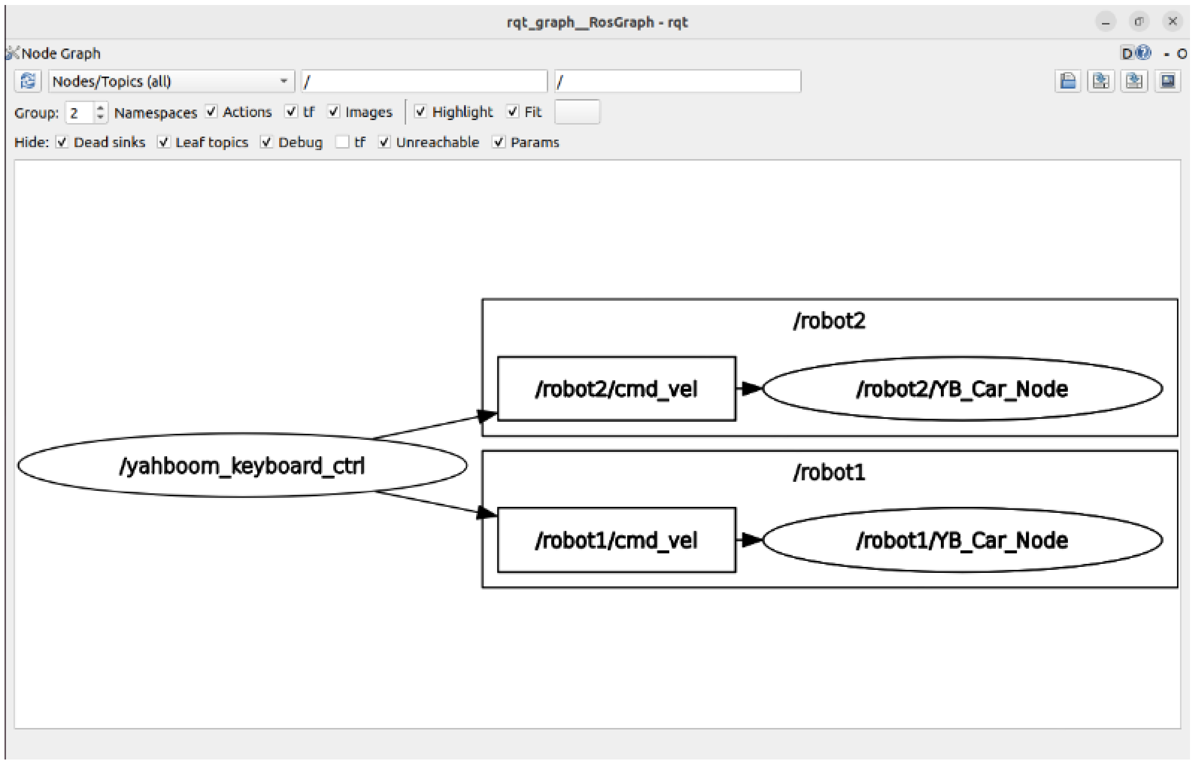
multi_yahboom_keyboard.py

```
#Create two velocity publishers
self.pub_robot1 = self.create_publisher(Twist,'/robot1/cmd_vel',1000)
self.pub_robot2 = self.create_publisher(Twist,'/robot2/cmd_vel',1000)
#Get key event
def getKey(self):
    tty.setraw(sys.stdin.fileno())
    rlist, _, _ = select.select([sys.stdin], [], [], 0.1)
    if rlist: key = sys.stdin.read(1)
else: key = ''
termios.tcsetattr(sys.stdin, termios.TCSADRAIN, self.settings)
return key
#Analyze key events
while (1):
    key = yahboom_keyboard.getKey()
    if key=="t" or key == "T": xspeed_switch = not xspeed_switch
    elif key == "s" or key == "S":
    print ("stop keyboard control: {}".format(not stop))
    stop = not stop
    if key in moveBindings.keys():
        x = moveBindings[key][0]
        th = moveBindings[key][1]
        count = 0
    elif key in speedBindings.keys():
        speed = speed * speedBindings[key][0]
        turn = turn * speedBindings[key][1]
        count = 0
        if speed > yahboom_keyboard.linenar_speed_limit:
            speed = yahboom_keyboard.linenar_speed_limit
            print("Linear speed limit reached!")
            if turn > yahboom_keyboard.angular_speed_limit:
                turn = yahboom_keyboard.angular_speed_limit
                print("Angular speed limit reached!")
                print(yahboom_keyboard.vels(speed, turn))
                if (status == 14): print(msg)
                status = (status + 1) % 15
            elif key == ' ': (x, th) = (0, 0)
        else:
            count = count + 1
            if count > 4: (x, th) = (0, 0)
            if (key == '\x03'): break
#Publish car speed
yahboom_keyboard.pub_robot1.publish(robot1_twist)
yahboom_keyboard.pub_robot2.publish(robot2_twist)
```

6、View node communication diagram

Select one of the two virtual machines, open the terminal and enter the command.

```
ros2 run rqt_graph rqt_graph
```



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