Keyboard Control

This lesson uses the Raspberry Pi as an example. For Raspberry Pi and Jetson Nano , you need to open a terminal on the host computer and enter the command to enter the Docker container. Once inside the Docker container, enter the commands mentioned in this lesson in the terminal. For instructions on entering the Docker container from the host computer, refer to **[01. Robot Configuration and Operation Guide] -- [5.Enter Docker (For JETSON Nano and RPi 5)]**. For Orin , simply open a terminal and enter the commands mentioned in this lesson.

1. Program Functionality

Power on the chassis and run the keyboard control program. You can control the car's movements with the keyboard. The handle also has functions such as controlling the buzzer and servos.

2. Enable Keyboard Control

2.1. Startup Command

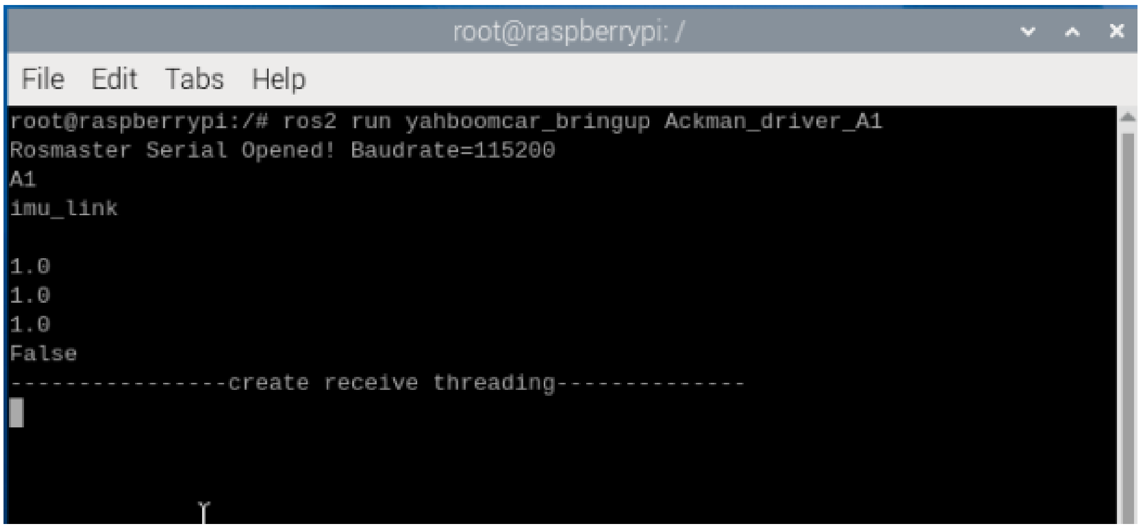
For the Raspberry Pi 5 controller, you must first enter the Docker container. For the Orin controller, this is not necessary.

Enter the Docker container (for steps, see [Docker Course] --- [4. Docker Startup Script]).

All the following commands must be executed from the Docker terminal within the same Docker container.**(For steps, see [Docker Course] --- [3. Docker Submission and Multi-Terminal Access]).**

To enable chassis data, enter the following command in the terminal:

```
ros2 run yahboomcar_bringup Ackman_driver_A1
```

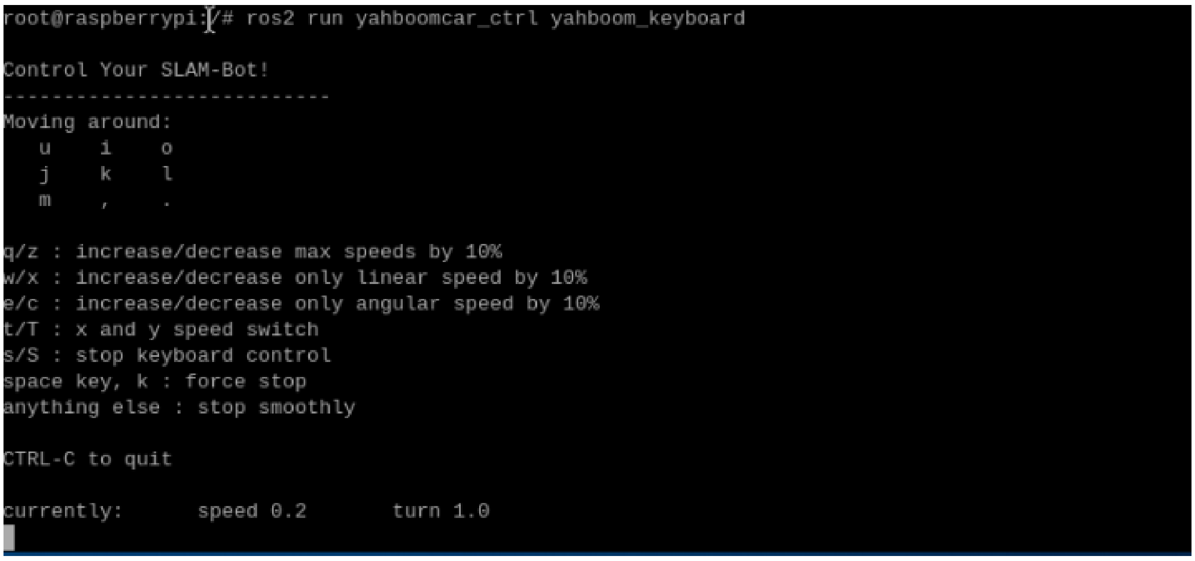


```
root@raspberrypi:/# ros2 run yahboomcar_bringup Ackman_driver_A1
Rosmaster Serial Opened! Baudrate=115200
A1
imu_link

1.0
1.0
1.0
False
----------------create receive threading-------------
```

Enter the following command in the terminal to start the keyboard control program:

```
ros2 run yahboomcar_ctrl yahboom_keyboard
```



```
root@raspberrypi:/# ros2 run yahboomcar_ctrl yahboom_keyboard

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

q/z : increase/decrease max speeds by 10%
w/x : increase/decrease only linear speed by 10%
e/c : increase/decrease only angular speed by 10%
t/T : x and y speed switch
s/S : stop keyboard control
space key, k : force stop
anything else : stop smoothly

CTRL-C to quit

currently:      speed 0.2       turn 1.0
```

Keyboard key descriptions are as follows:

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]

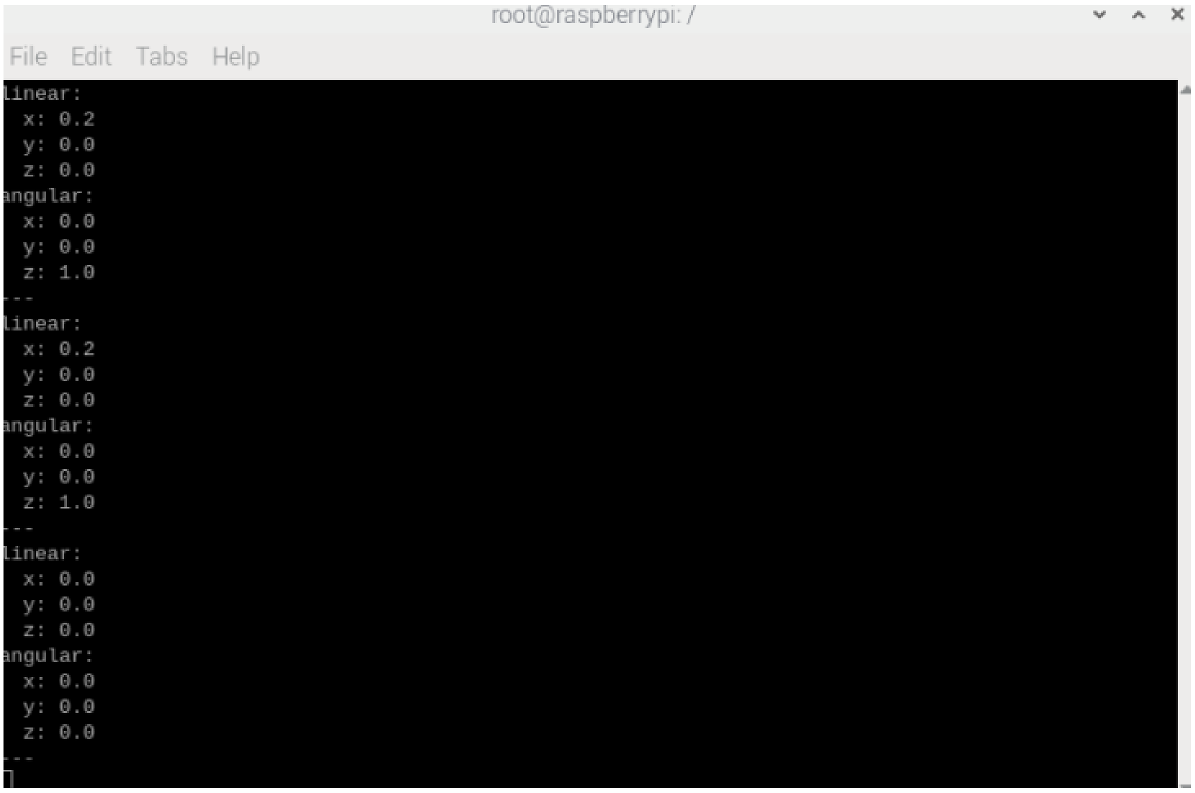
That is, press [i] to move forward, press [,] to move backward, press [l] to rotate right, press [j] to rotate left, and so on.

Speed Control

Keys	Speed Change	Keys	Speed Change
【q】	Increase both linear and angular speeds by 10%	【z】	Decrease both linear and angular speeds by 10%
【w】	Increase only linear speed by 10%	【x】	Decrease only linear speed by 10%
【e】	Increase only angular speed by 10%	【c】	Decrease only angular speed by 10%
【t】	Switch linear speed between X and Y axes	【s】	Stop keyboard control

You can open another terminal to view real-time /cmd_cel speed topic data.

```
ros2 topic echo /cmd_vel
```



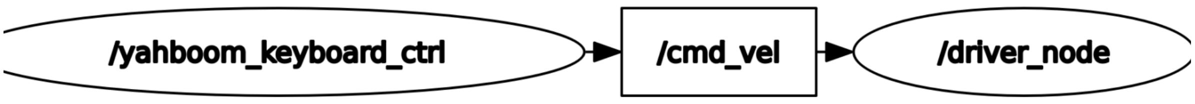
```
linear:
  x: 0.2
  y: 0.0
  z: 0.0
angular:
  x: 0.0
  y: 0.0
  z: 1.0
---
linear:
  x: 0.2
  y: 0.0
  z: 0.0
angular:
  x: 0.0
  y: 0.0
  z: 1.0
---
linear:
  x: 0.0
  y: 0.0
  z: 0.0
angular:
  x: 0.0
  y: 0.0
  z: 0.0
---
```

Because the robot chassis can only move within a two-dimensional plane, only linear-x (x-axis velocity), linear-y (y-axis velocity), and angular-z (z-axis angular velocity) are issued when controlling the robot via the keyboard.

3. View the Node Relationship Graph

Open a terminal and enter the command:

```
ros2 run rqt_graph rqt_graph
```



From the node relationship diagram, we can see:

yahboom_keyboard_ctrl: Controls the robot chassis by publishing the **/cmd_vel** topic

/driver_node: The robot chassis node subscribes to the **/cmd_vel** topic and communicates with the ROS expansion board to determine the speed of each wheel, thereby controlling the car's movement.

4. Source Code Analysis

Source Code Path:

```
~/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_ctrl/yahboomcar_ctrl/yahboom_k
eyboard.py
```

- The movement dictionary primarily stores characters related to direction control.

```
moveBindings = {
    'i': (1, 0),
    'o': (1, -1),
    'j': (0, 1),
    'l': (0, -1),
    'u': (1, 1),
    ',': (-1, 0),
    '.': (-1, 1),
    'm': (-1, -1),
    'I': (1, 0),
    'O': (1, -1),
    'J': (0, 1),
    'L': (0, -1),
    'U': (1, 1),
    'M': (-1, -1),
}
```

- The speed dictionary mainly stores the characters related to speed control

```
speedBindings = {
    'Q': (1.1, 1.1),
    'Z': (.9, .9),
    'W': (1.1, 1),
    'X': (.9, 1),
    'E': (1, 1.1),
    'C': (1, .9),
    'q': (1.1, 1.1),
    'z': (.9, .9),
    'w': (1.1, 1),
    'x': (.9, 1),
    'e': (1, 1.1),
    'c': (1, .9),
}
```

Get the current key information

```
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
```

Determine the key value and publish the /cmd_vel speed topic

```
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
    if xspeed_switch: twist.linear.x = speed * x
    else: twist.linear.y = speed * x
    twist.angular.z = turn * th
    if not stop: yahboom_keyboard.pub.publish(twist)
    if stop:yahboom_keyboard.pub.publish(Twist())
```