QR code motion control

1. Gameplay Introduction

This course primarily utilizes the robot's camera to capture camera images, recognize QR codes, and control the robot's movements based on the QR code information.

For Raspberry Pi and Jetson-Nano boards, 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] --[4.Enter Docker (For JETSON Nano and RPi 5)]**.

For RDKX5 and Orin boards, simply open a terminal and enter the commands mentioned in this lesson.

2. Program Code Reference Path

```
~/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_astra/yahboomcar_astra
```

3. Program Startup

3.1. Startup Command

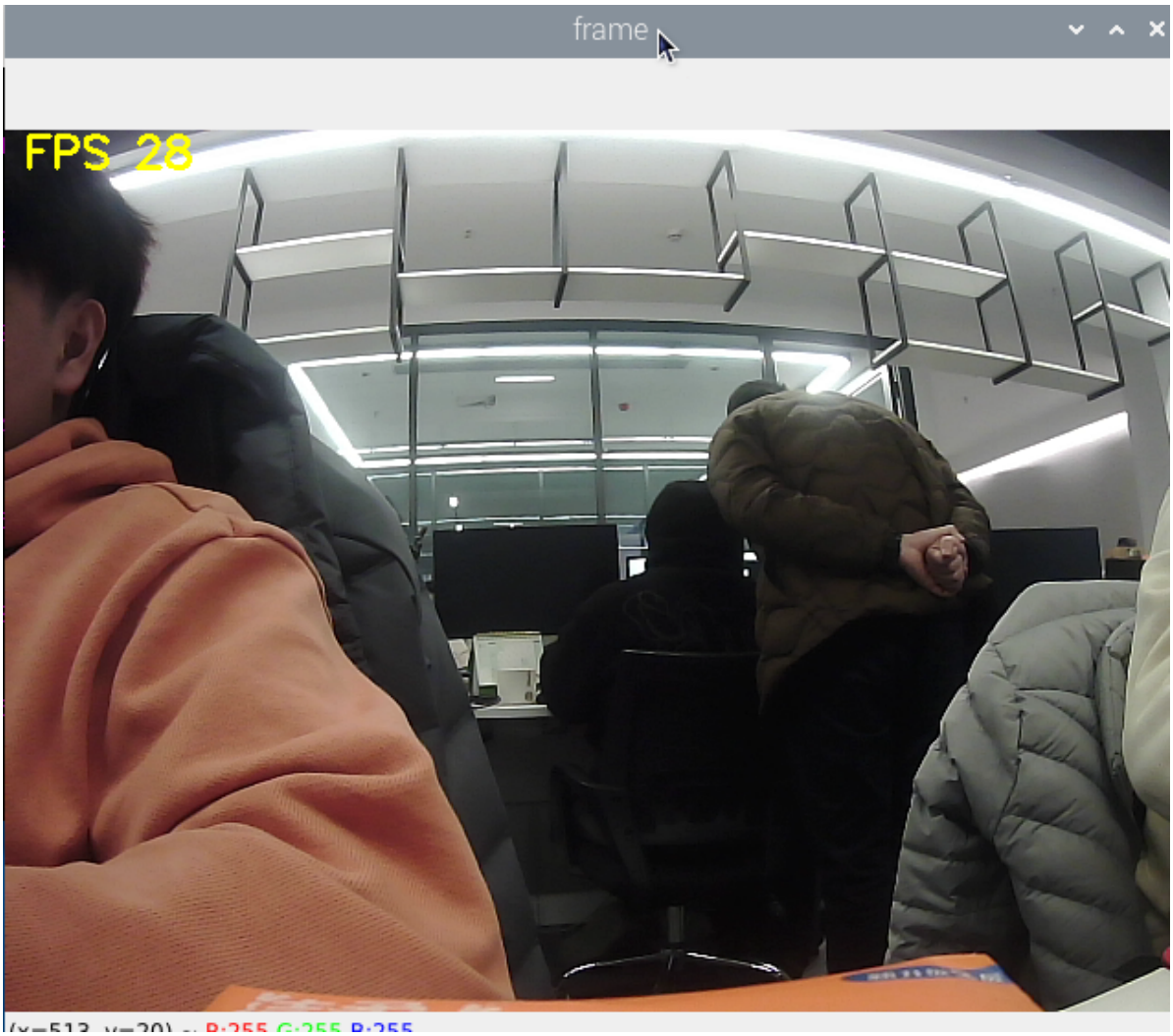
First, enter the following command in the terminal to start the camera:

```
#usb camera
ros2 launch usb_cam camera.launch.py
#nuwa camera
ros2 launch ascamera hp60c.launch.py
```

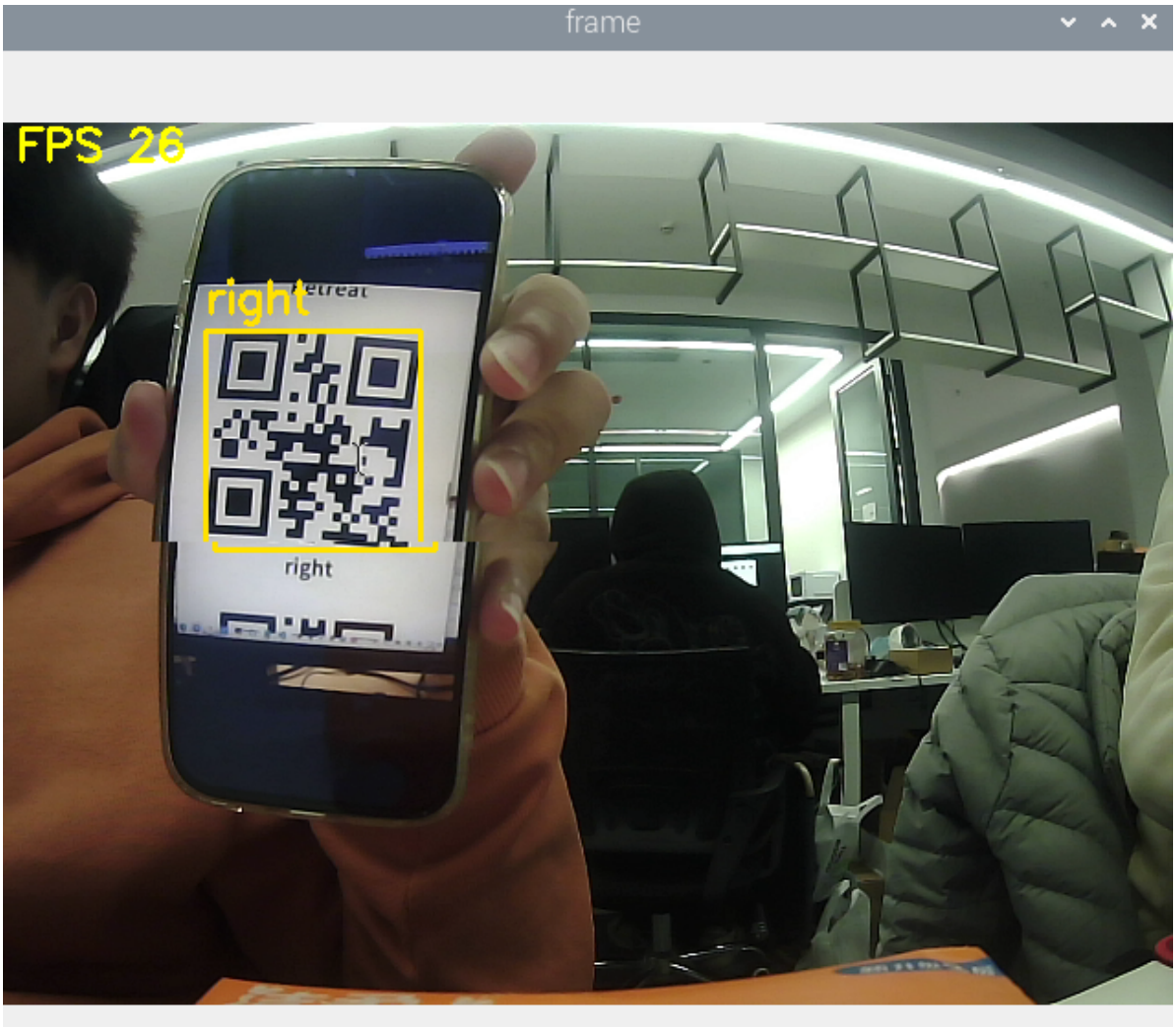
After successfully starting the camera, open another terminal and enter the following command:

```
#chassis driver
ros2 launch yahboomcar_bringup yahboomcar_bringup_M1_launch.py
#QR code control
ros2 run yahboomcar_astra qrCtrl
```

The camera image is successfully displayed



Enable QR code recognition and execute commands. Currently, this example recognizes QR codes. The following information is displayed: "forward" for forward, "back" for backward, "left" for left turn, "right" for right turn, and "stop" for stop.



Press q to turn off the camera.

4. Core Code

Import the QR code parsing library pybar

```
import pyzbar.pyzbar as pyzbar
from PIL import Image
```

Parse a grayscale image and extract the QR code information and image location. If there is no QR code in the image, the information will be None.

```
def detect_qrcode(image):
    # 转为灰度图像 Convert to grayscale image
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    barcodes = pyzbar.decode(gray)
    for barcode in barcodes:
        # 提取二维码的数据和边界框的位置 The data of the QR code and the position of
the bounding box are extracted
        (x, y, w, h) = barcode.rect
        barcodeData = barcode.data.decode("utf-8")
        barcodeType = barcode.type
        # print("[INFO] Found {} barcode: {}".format(barcodeType, barcodeData))
        car_control(barcodeData)
        return barcodeData, (x, y, w, h)
    return None, (0, 0, 0, 0)
```

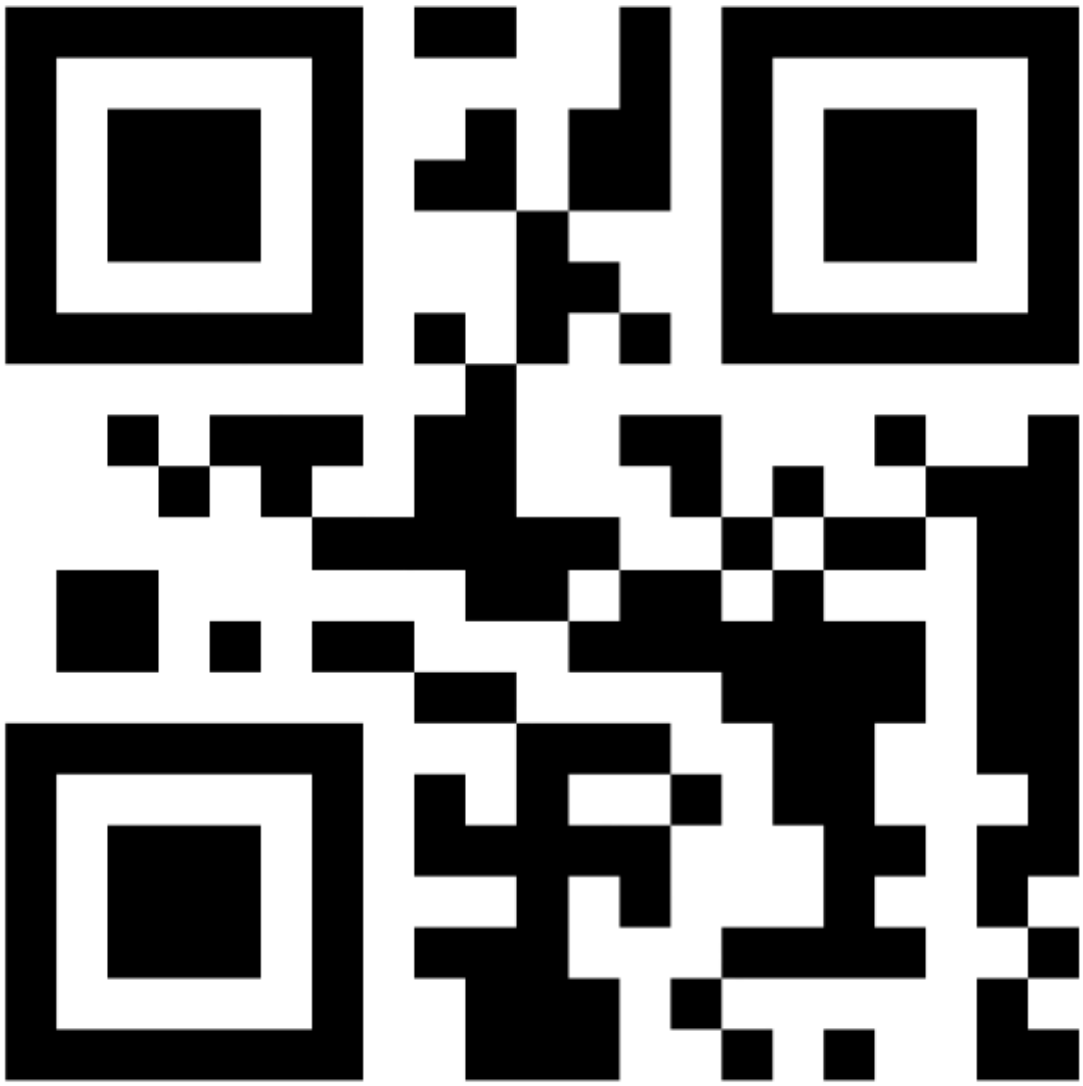
Control the robot's movement according to the string command of info.

```
def robot_action(self,data):
    if data == "forward":
        self.pub_vel(0.1,0.0,0.0)
        sleep(0.5)

    elif data == "back":
        self.pub_vel(-0.1,0.0,0.0)
        sleep(0.5)

    elif data == "left":
        self.pub_vel(0.1,0.0,1.0)
        sleep(0.5)

    elif data == "right":
        self.pub_vel(0.1,0.0,-1.0)
        sleep(0.5)

    elif data == "stop":
        self.pub_vel(0.0,0.0,0.0)
        sleep(0.5)
```

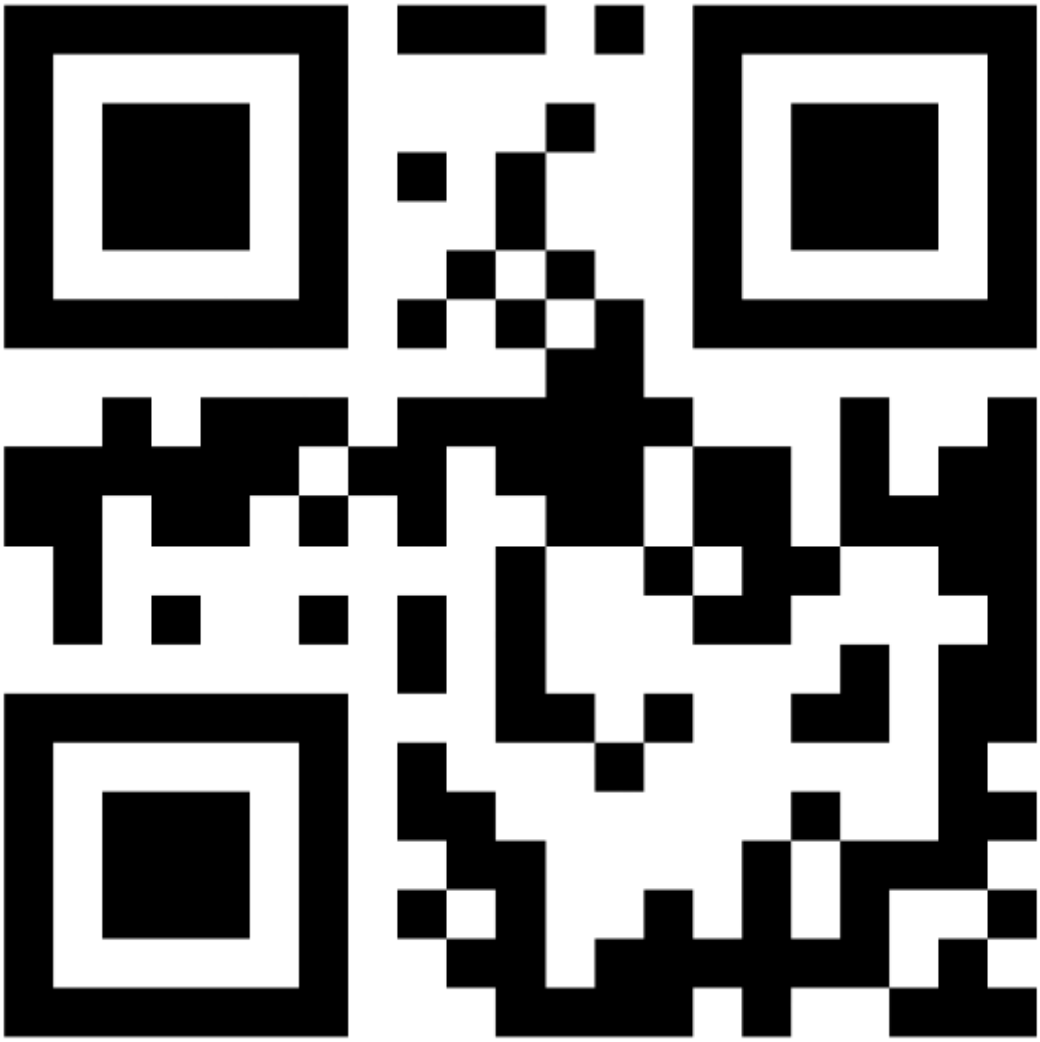
Image processing program

```
def image_callback(self, msg):
    frame = self.bridge.imgmsg_to_cv2(msg, desired_encoding='bgr8')

    payload, (x, y, w, h) = self.detect_qrcode(frame)
    if payload != None:
        cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 225, 255), 2)
        cv2.putText(frame, payload, (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.8,
(0, 225, 255), 2)
        self.robot_action(payload)
    else:
        self.pub_vel(0.0,0.0,0.0)

    cv2.imshow('frame', frame)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        self.get_logger().info("Exiting program")
        cv2.destroyAllWindows()
        rclpy.shutdown()
        exit(0)
```

QR code image:



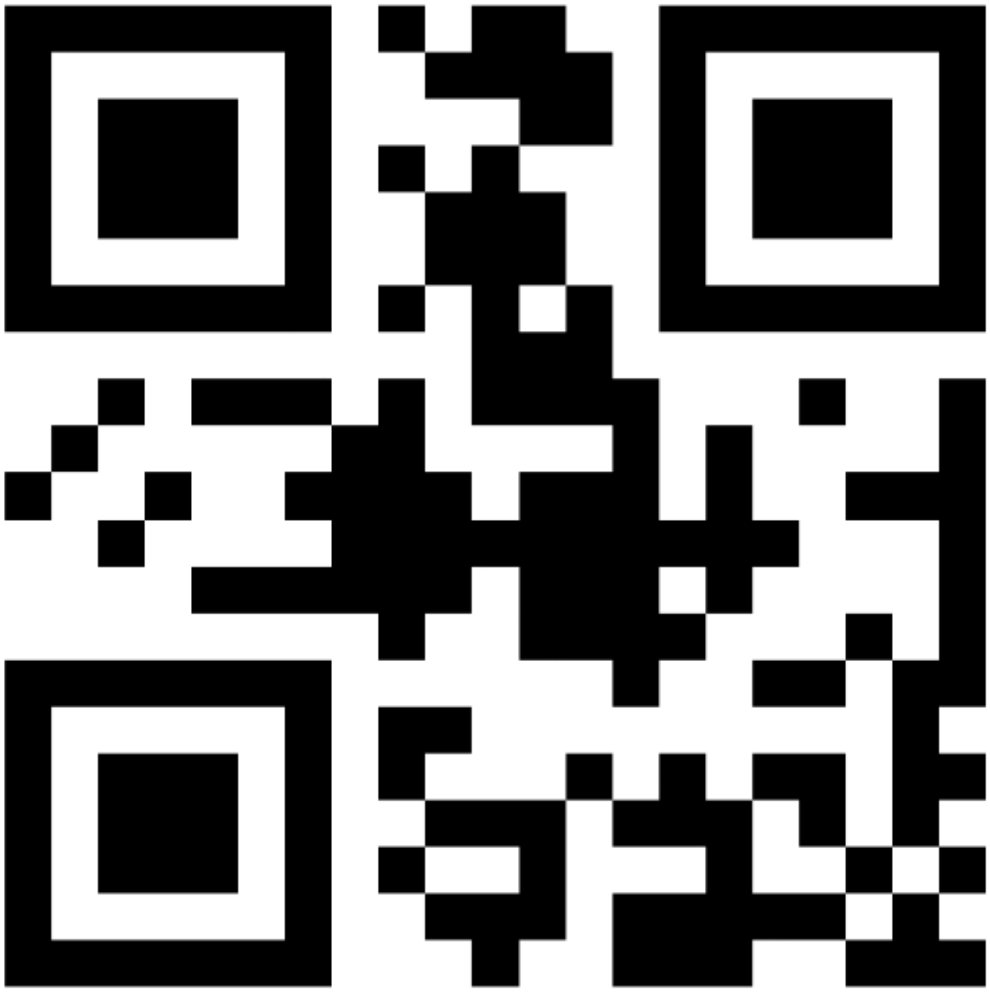
forward



back

left

right



stop