

### 3.1.6 QR recognition+Movement

As shown in the figure below, the five QR codes correspond to different functions and can be used to control the movement of the robot.



forward



back



left



right



stop

Code path:

[/home/pi/Yahboom\\_Project/2.AI\\_Visual\\_course/06.QR\\_code\\_move.ipynb](/home/pi/Yahboom_Project/2.AI_Visual_course/06.QR_code_move.ipynb)

```

#bgr8 to jpeg format
import enum
import cv2

def bgr8_to_jpeg(value, quality=75):
    return bytes(cv2.imencode('.jpg', value)[1])

# import the necessary packages
#import simple_barcode_detection
import cv2
import numpy as np
import pyzbar.pyzbar as pyzbar
from PIL import Image
import ipywidgets.widgets as widgets

#-*- coding:UTF-8 -*-
import RPi.GPIO as GPIO
import time

#Set GPIO port to BCM coding mode
GPIO.setmode(GPIO.BCM)

#Ignore the warning message
GPIO.setwarnings(False)

#Define motor pin
IN1 = 20
IN2 = 21
IN3 = 19
IN4 = 26
ENA = 16
ENB = 13

image_widget = widgets.Image(format='jpeg', width=320, height=240)
display(image_widget) #display camera video

#Motor pin initialization operation
def motor_init():
    global pwm_ENA
    global pwm_ENB
    global delaytime
    GPIO.setup(ENA,GPIO.OUT,initial=GPIO.HIGH)
    GPIO.setup(IN1,GPIO.OUT,initial=GPIO.LOW)
    GPIO.setup(IN2,GPIO.OUT,initial=GPIO.LOW)
    GPIO.setup(ENB,GPIO.OUT,initial=GPIO.HIGH)

```

```
GPIO.setup(IN3,GPIO.OUT,initial=GPIO.LOW)
GPIO.setup(IN4,GPIO.OUT,initial=GPIO.LOW)
#设置 pwm 引脚和频率为 2000hz
pwm_ENA = GPIO.PWM(ENA, 2000)
pwm_ENB = GPIO.PWM(ENB, 2000)
pwm_ENA.start(0)
pwm_ENB.start(0)
```

#car advance

```
def run(delaytime):
    GPIO.output(IN1, GPIO.HIGH)
    GPIO.output(IN2, GPIO.LOW)
    GPIO.output(IN3, GPIO.HIGH)
    GPIO.output(IN4, GPIO.LOW)
    pwm_ENA.ChangeDutyCycle(80)
    pwm_ENB.ChangeDutyCycle(80)
    time.sleep(delaytime)
```

#car back

```
def back(delaytime):
    GPIO.output(IN1, GPIO.LOW)
    GPIO.output(IN2, GPIO.HIGH)
    GPIO.output(IN3, GPIO.LOW)
    GPIO.output(IN4, GPIO.HIGH)
    pwm_ENA.ChangeDutyCycle(80)
    pwm_ENB.ChangeDutyCycle(80)
    time.sleep(delaytime)
```

#car turn left

```
def left(delaytime):
    GPIO.output(IN1, GPIO.LOW)
    GPIO.output(IN2, GPIO.LOW)
    GPIO.output(IN3, GPIO.HIGH)
    GPIO.output(IN4, GPIO.LOW)
    pwm_ENA.ChangeDutyCycle(80)
    pwm_ENB.ChangeDutyCycle(80)
    time.sleep(delaytime)
```

#car turn right

```
def right(delaytime):
    GPIO.output(IN1, GPIO.HIGH)
    GPIO.output(IN2, GPIO.LOW)
    GPIO.output(IN3, GPIO.LOW)
    GPIO.output(IN4, GPIO.LOW)
```

```
pwm_ENA.ChangeDutyCycle(80)
pwm_ENB.ChangeDutyCycle(80)
time.sleep(delaytime)

#czar spin left
def spin_left(delaytime):
    GPIO.output(IN1, GPIO.LOW)
    GPIO.output(IN2, GPIO.HIGH)
    GPIO.output(IN3, GPIO.HIGH)
    GPIO.output(IN4, GPIO.LOW)
    pwm_ENA.ChangeDutyCycle(80)
    pwm_ENB.ChangeDutyCycle(80)
    time.sleep(delaytime)

#car spin right
def spin_right(delaytime):
    GPIO.output(IN1, GPIO.HIGH)
    GPIO.output(IN2, GPIO.LOW)
    GPIO.output(IN3, GPIO.LOW)
    GPIO.output(IN4, GPIO.HIGH)
    pwm_ENA.ChangeDutyCycle(80)
    pwm_ENB.ChangeDutyCycle(80)
    time.sleep(delaytime)

#car stop
def brake(delaytime):
    GPIO.output(IN1, GPIO.LOW)
    GPIO.output(IN2, GPIO.LOW)
    GPIO.output(IN3, GPIO.LOW)
    GPIO.output(IN4, GPIO.LOW)
    pwm_ENA.ChangeDutyCycle(80)
    pwm_ENB.ChangeDutyCycle(80)
    time.sleep(delaytime)

def detect_control(info):
    if info == "forward":
        run(1)
        brake(1)
    elif info == "back":
        back(1)
        brake(1)
    elif info == "left":
        left(1)
        brake(1)
    elif info == "right":
```

```

        right(1)
        brake(1)
    elif info == "brake":
        brake(1)

# Define the parse QR code interface
def decodeDisplay(image):
    barcodes = pyzbar.decode(image)
    for barcode in barcodes:
        # Extract the position of the bounding box of the QR code
        # Draw the bounding box of the barcode in the image
        (x, y, w, h) = barcode.rect
        cv2.rectangle(image, (x, y), (x + w, y + h), (225, 225, 225), 2)

        # Extract the QR code data as a byte object, so if we want to output the
        image, you need to convert it to a string
        barcodeData = barcode.data.decode("utf-8")
        barcodeType = barcode.type

        # Draws the data and barcode type of the barcode on the image
        text = "{} {}".format(barcodeData, barcodeType)
        cv2.putText(image, text, (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (225,
225, 225), 2)

        # Print the data and barcode type of the barcode on the terminal
        print("[INFO] Found {} barcode: {}".format(barcodeType, barcodeData))
        detect_control(barcodeData)
    return image

def detect():
    camera = cv2.VideoCapture(0)
    camera.set(3, 320)
    camera.set(4, 240)
    camera.set(5, 120) #Set frame rate
    # fourcc = cv2.VideoWriter_fourcc(*"MPEG")
    camera.set(cv2.CAP_PROP_FOURCC, cv2.VideoWriter_fourcc('M', 'J', 'P', 'G'))
    camera.set(cv2.CAP_PROP_BRIGHTNESS, 40) #Set brightness -64 - 64 0.0
    camera.set(cv2.CAP_PROP_CONTRAST, 50) #Set contrast -64 - 64 2.0
    camera.set(cv2.CAP_PROP_EXPOSURE, 156) #Set exposure 1.0 - 5000 156.0
    ret, frame = camera.read()
    image_widget.value = bgr8_to_jpeg(frame)
    while True:
        # Read frame currently
        ret, frame = camera.read()
        # To Grayscale image

```

```

gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
im = decodeDisplay(gray)

cv2.waitKey(5)
image_widget.value = bgr8_to_jpeg(im)
# If you press q, you will be out of the loop
if cv2.waitKey(10) & 0xFF == ord('q'):
    break
camera.release()
cv2.destroyAllWindows()

```

```

while 1:
    detect()

```

After run above program, we can realize QR code control car movement. As shown below.

