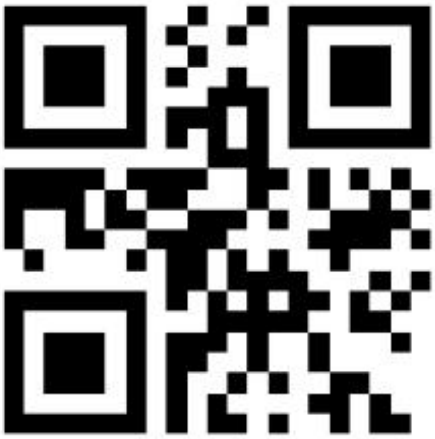
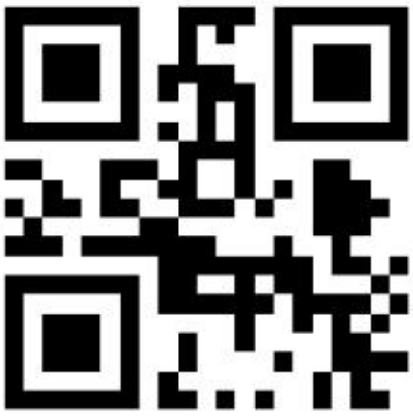
3.1.6 QR recognition+Movement

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.

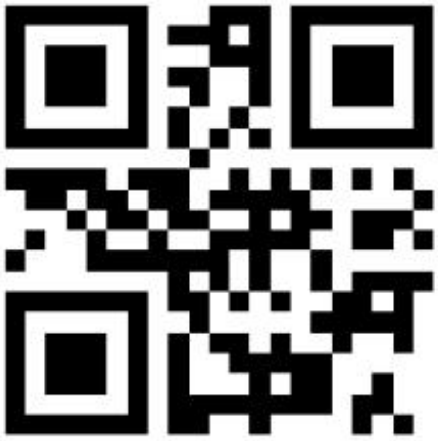
forword



back



left



right

stop

Code path:

/home/pi/Yahboom_Project/3.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 library and show camera display component
# 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

# Underlying drive method
from Raspblock import Raspblock
robot = Raspblock()

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

```
# Define the recognition motion function
def detect_control(info):
    if info == "forward":
        robot.Speed_axis_Yawhold_control(0, 2)      # Advance
    elif info == "back":
        robot.Speed_axis_Yawhold_control(0, -2)    # Back
    elif info == "left":
        robot.Speed_axis_Yawhold_control(-2, 0)    # Left translation
    elif info == "right":
        robot.Speed_axis_Yawhold_control(2, 0)      # Right translation
    else:
        robot.Speed_axis_Yawhold_control(0, 0)      # Stop
```

```
# 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.

```
[2]: # 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

     #Underlying drive method
     from Raspblock import Raspblock
     robot = Raspblock()

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

```
serial Open!
```


```
[5]: while 1:
         detect()
```

```
[INFO] Found QRCODE barcode: forward
[INFO] Found QRCODE barcode: forward
[INFO] Found QRCODE barcode: forward
[INFO] Found QRCODE barcode: forward
[INFO] Found QRCODE barcode: forward
[INFO] Found QRCODE barcode: forward
[INFO] Found QRCODE barcode: forward
[INFO] Found QRCODE barcode: forward
[INFO] Found QRCODE barcode: forward
[INFO] Found QRCODE barcode: forward
[INFO] Found QRCODE barcode: forward
```