3、QR code creation and recognition

This tutorial takes usb no-drive camera as an example. All codes run in docker. When entering docker, you need to mount the camera. For details, please refer to docker tutorial [5、Enter the docker Container].

1、QR code

1.1、Introduction to QR codes

QR code is a type of two-dimensional barcode, which comes from the abbreviation of "Quick Response" in English, meaning "quick response". It originates from the inventor's hope that QR code can quickly decode its content. QR codes not only have large information capacity, high reliability, and low cost, but also can represent various textual information such as Chinese characters and images. They have strong confidentiality and anti-counterfeiting capabilities, and are very convenient to use. More importantly, the QR code technology is open source.

1.2、Structure of QR codes

picture	Parse
	Positioning markings indicate the direction of the QR code.
	Alignment markings If the QR code is large, these additional elements help with positioning.
	pattern With these lines, the scanner can identify how big the matrix is.
	Version information (Version information) here specifies the version number of the QR code in use. There are currently 40 different version numbers of the QR code. Version numbers for the sales industry are usually 1-7.
	Format information Format patterns contain information about fault tolerance and data mask patterns and make scanning codes easier.
	Data and error correction keys These modes hold the actual data.
	Quiet zone This zone is very important for the scanner, its role is to separate itself from the surrounding.

1.3、Characteristics of QR codes

The data value in the QR code contains duplicate information (redundant values).Therefore, even if up to 30% of the QR code structure is destroyed without affecting the readability of the QR code. The QR code has a storage space of 7089 bits or 4296 characters, including Punctuation and special characters, which can be written into the QR code. In addition to numbers and characters, words and phrases (such as web addresses) can also be encoded. As more data is added to the QR code, the code size increases and the code structure becomes more complex.

1.4、QR code creation and recognition

1)、Source code path

```
/root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_visual/simple_qrcode
```

2)、Installation package

```
python3 -m pip install qrcode pyzbar
sudo apt-get install libzbar-dev
```

NOTE: The factory Docker image has been installed.

3)、Create QRcode_ Create. py

Enter Docker and open Terminal Input，

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_visual/simple_qrcode/
python QRcode_Create.py
```

After the program runs, you will be prompted to enter the generated content, and the Enter key key will confirm the content. Here, taking the creation of the "yahboom" string as an example,
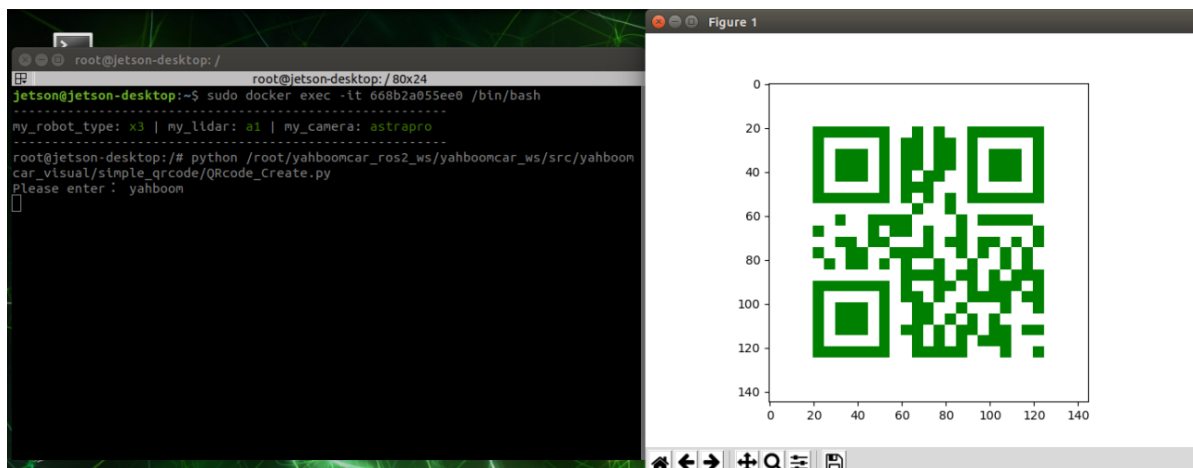
The QR code that appears on the right, take out your phone and try scanning it. The scanning result will be the characters of "yahboom".

Source code parsing，

```
#Create qrcode object
qr = qrcode.QRCode(
        version=1,
        error_correction=qrcode.constants.ERROR_CORRECT_H,
```

```python
        box_size=5,
        border=4,)
#Meaning of each parameter
'''version: An integer with a value of 1-40 controls the size of the QR code (the
minimum value is 1, which is a 12 × 12 matrix). If you want the program to
automatically determine, set the value to "None" and use the "fit" parameter.
error_correction: Control the error correction function of the QR code. The
following four constants can be taken as values.
        ERROR_CORRECT_L: Approximately 7% or less of errors can be corrected.
        ERROR_CORRECT_M（default认）: Approximately 15% or less of errors can be
corrected.
        ROR_CORRECT_H: Approximately 30% or less of errors can be corrected.
box_size: Control the number of pixels contained in each small grid in the QR
code.
border: Control the number of cells included in the border (the distance between
the QR code and the image boundary) (default is 4, which is the minimum value
specified by relevant standards)
#Adding logo to qrcode QR code
my_file = Path(logo_path)
if my_file.is_file(): img = add_logo(img, logo_path)
#Add data
qr.add_data(data)
# Fill in data
# fill data
qr.make(fit=True)
# Generate images
# generate images
img = qr.make_image(fill_color="green", back_color="white")
```
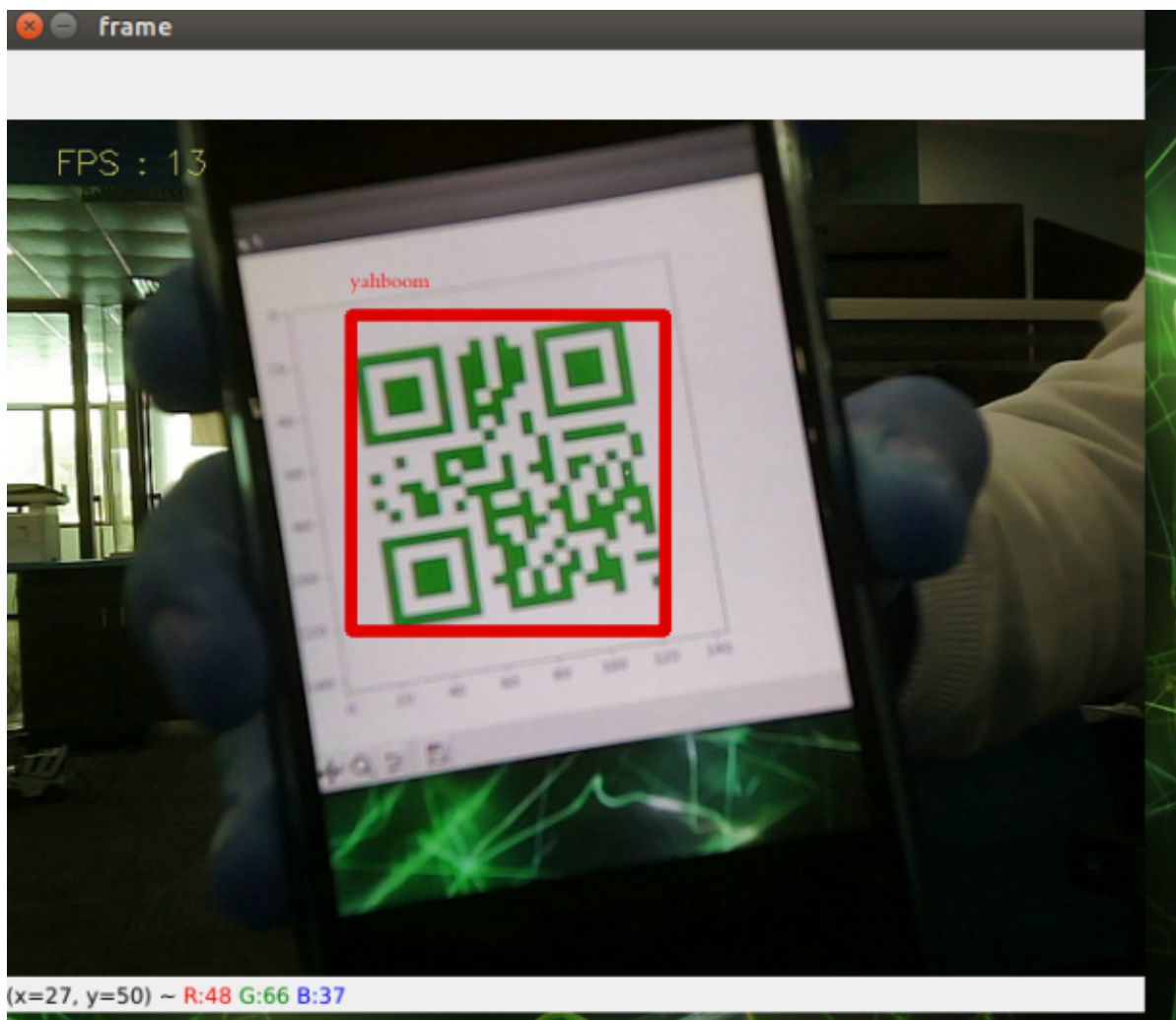
4)、Identify QRcode_ Parsing.py

Enter Docker and open Terminal Input,

```
cd /root/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_visual/simple_qrcode/
python QRcode_Parsing.py
```

After the program runs, we place the QR code in front of the camera. The program will recognize the content of the QR code, mark it on the image, and print the recognized content on the terminal。

Source code parsing，

```python
def decodeDisplay(image, font_path):
    gray = cv.cvtColor(image, cv.COLOR_BGR2GRAY)
    # The output Chinese characters need to be converted to Unicode encoding
first
    barcodes = pyzbar.decode(gray)
    for barcode in barcodes:
        # Extract the position of the bounding box for the QR code
        (x, y, w, h) = barcode.rect
        # Draw the bounding box of the barcode in the image
        cv.rectangle(image, (x, y), (x + w, y + h), (225, 0, 0), 5)
        encoding = 'UTF-8'
        # To draw it, you need to first convert it into a string
        barcodeData = barcode.data.decode(encoding)
        barcodeType = barcode.type
        # Draw data and types on the image
        pilimg = Image.fromarray(image)
        # Create a brush
        draw = ImageDraw.Draw(pilimg)
        # Parameter 1: Font file path, Parameter 2: Font size
        fontStyle = ImageFont.truetype(font_path, size=12, encoding=encoding)
```

```python
    # Parameter 1: Print coordinates, Parameter 2: Text, Parameter 3: Font color,
Parameter 4: Font
    draw.text((x, y - 25), str(barcode.data, encoding), fill=(255, 0,
0),font=fontStyle)
    # PIL image to cv2 image
    image = cv.cvtColor(np.array(pilimg), cv.COLOR_RGB2BGR)
    # Print barcode data and barcode type to the terminal
    print("[INFO] Found {} barcode: {}".format(barcodeType, barcodeData))
    return image
```