5.QR code creation and recognition

1. QR code

1.1. Introduction to QR code

QR code is a type of two-dimensional barcode. QR comes from the abbreviation of "Quick Response" in English, which means quick response. It comes from the inventor's hope that the content of QR code can be decoded quickly. QR code not only has large information capacity, high reliability, and low cost, but can also represent various text information such as Chinese characters and images. It has strong confidentiality and anti-counterfeiting and is very convenient to use. What's more important is that the QR code technology is open source.

1.2. Structure of QR code

图片	解析
	**定位标识**（Positioning markings）标明二维码的方向。
	**对齐标记**（Alignment markings）如果二维码很大，这些附加元素帮助定位。
	**计算模式**（Timing pattern）通过这些线，扫描器可以识别矩阵有多大。
	**版本信息**（Version information）这里指定正在使用的QR码的版本号，目前有QR码有40个不同的版本号。 用于销售行业的的版本号通常为1-7。
	**格式信息**（Format information）格式模式包含关于容错和数据掩码模式的信息，并使得扫描代码更加容易。
	**数据和错误校正值**（Data and error correction keys）这些模式保存实际数据。
	**宁静区域**（Quiet zone）这个区域对于扫描器来说非常重要，它的作用就是将自身与周边的进行分离。

1.3. Features of QR code

The data values in the QR code contain repeated information (redundant values). Therefore, even if up to 30% of the QR code structure is destroyed, the readability of the QR code is not affected. The storage space of QR code is up to 7089 bits or 4296 characters, including punctuation marks and special characters, which can be written into 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
```

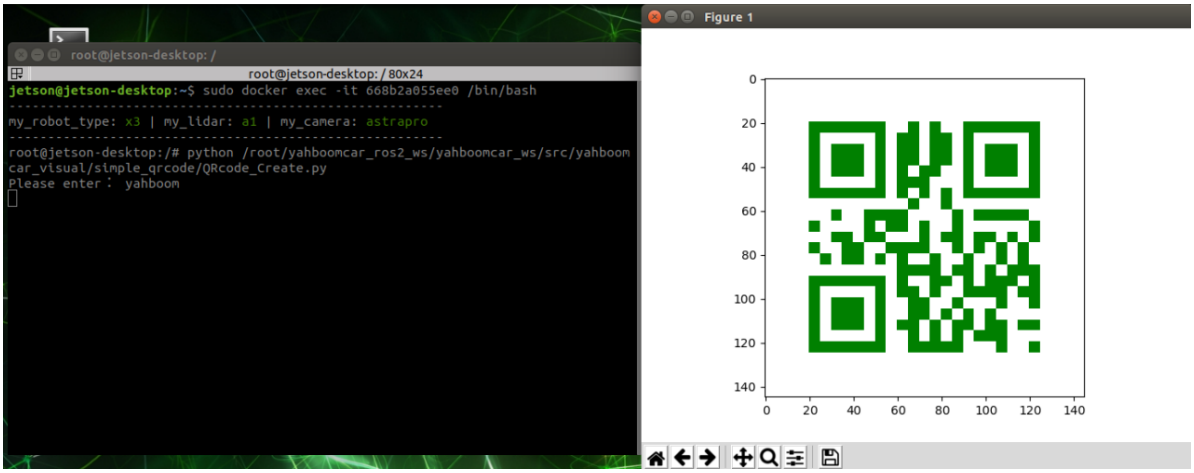
The factory docker image has been installed.

3.Create QRcode_Create.py

Enter docker, open the terminal and enter,

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

After the program is run, you will be prompted to enter the generated content, and press Enter to confirm the content. Here we take creating the "yahboom" string as the content as an example,



Take out your phone and try to scan the QR code that appears on the right. The scan result will be the characters yahboom.

Source code analysis,

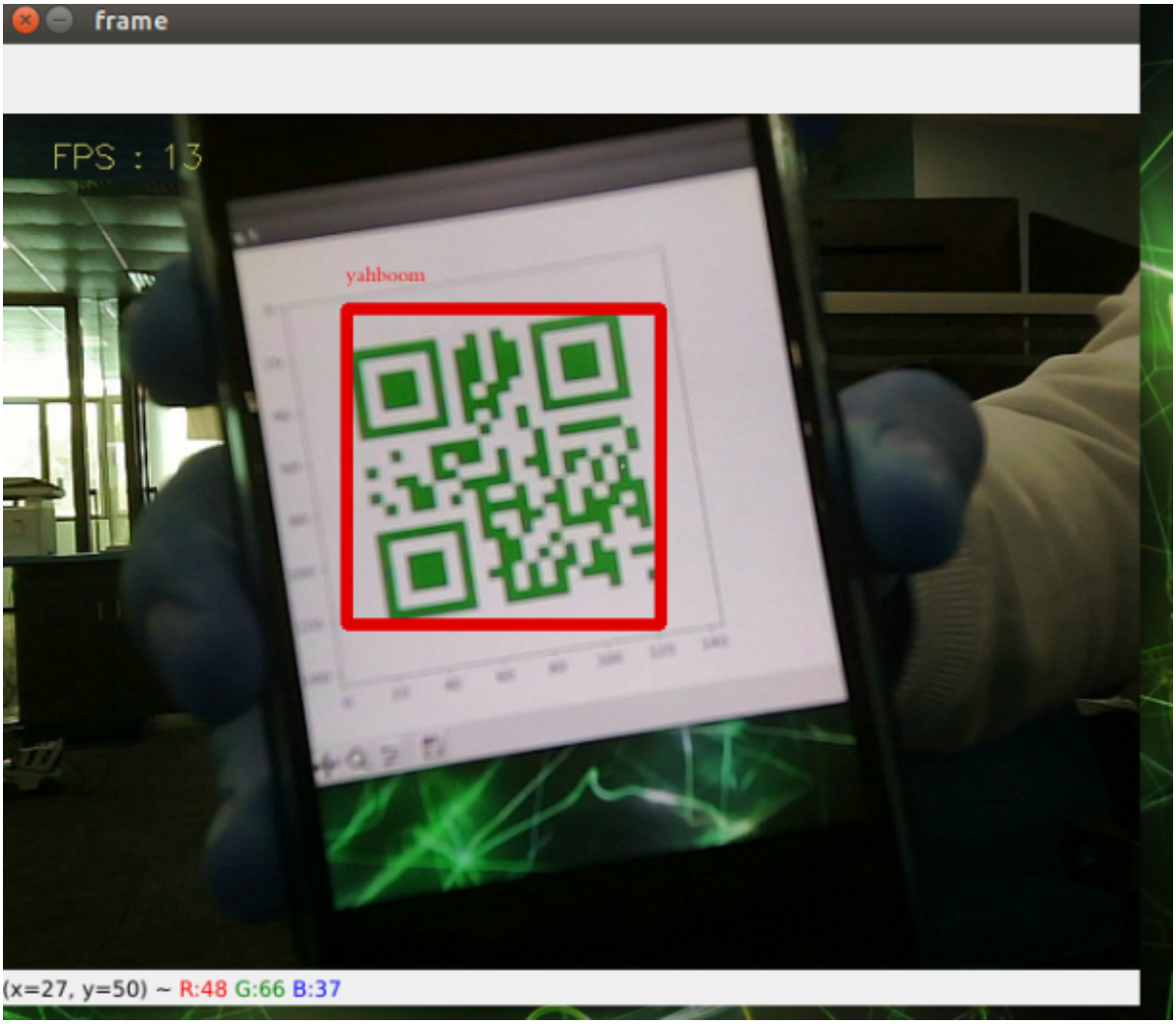
```
#Create qrcode object
qr = qrcode.QRCode(
        version=1,
        error_correction=qrcode.constants.ERROR_CORRECT_H,
```

```
        box_size=5,
        border=4,)
#Meanings of each parameter
'''version: An integer with a value from 1 to 40, controlling the size of the QR
code (the minimum value is 1, which is a 12×12 matrix).
             If you want the program to determine this automatically, set the
value to None and use the fit argument.
error_correction: Controls the error correction function of QR codes. Possible
values are the following 4 constants.
       ERROR_CORRECT_L: About 7% or less of errors can be corrected.
       ERROR_CORRECT_M (Default): Approximately 15% or less of errors can be
corrected.
       ROR_CORRECT_H: About 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 border) (the default is 4, which is the minimum value
specified by relevant standards)'''
#Add 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)
# Data input
# fill data
qr.make(fit=True)
# Generate pictures
# generate images
img = qr.make_image(fill_color="green", back_color="white")
```

4. Identify QRcode_Parsing.py

Enter docker, open the terminal and enter,

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



After the program is run, we place the QR code in front of the camera. The program will recognize the content of the QR code, mark it on the picture and print out the recognized content on the terminal.

Source code analysis,

```
def decodeDisplay(image, font_path):
    gray = cv.cvtColor(image, cv.COLOR_BGR2GRAY)
    # It is necessary to convert the output Chinese characters into Unicode
encoding form first.
    barcodes = pyzbar.decode(gray)
    for barcode in barcodes:
    # Extract the position of the bounding box of 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 convert it into a string first
    barcodeData = barcode.data.decode(encoding)
    barcodeType = barcode.type
    # Plot data and types on the image
    pilimg = Image.fromarray(image)
    # Create brushes
    draw = ImageDraw.Draw(pilimg)
    # Parameter 1: font file path, parameter 2: font size
    fontStyle = ImageFont.truetype(font_path, size=12, encoding=encoding)
```

```
    #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 terminal
    print("[INFO] Found {} barcode: {}".format(barcodeType, barcodeData))
    return image
```