8.1、Opencv application

8.1.1 Overview

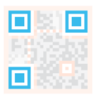
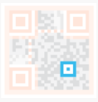
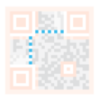
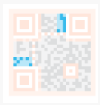
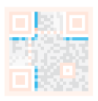
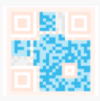
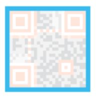
OpenCV is a cross-platform computer vision and machine learning software library based on a BSD license (open source) that can run on Linux, Windows, Android and MacOS operating systems. It provides interfaces to languages such as Python, Ruby, and MATLAB, and supports many general algorithms in image processing and computer vision.

8.1.2、QR code

1. Introduction of QR code

QR code not only has large information capacity, high reliability, and low cost, but it can also express a variety of text information such as Chinese characters and images. It has strong confidentiality and anti-counterfeiting and is very convenient to use.

2. QR code structure

Picture	Analyze
	Positioning markings: Indicate the direction of the QR code.
	Alignment markings: If the QR code is large, these additional elements help positioning.
	Timing pattern: Through these lines, the scanner the matrix can be can recognized.
	Version information: This refers to the version number of the QR code being used. There are currently 40 different versions of the QR code. The version number used in the sales industry is usually 1-7.
	Format information: It contains information about fault tolerance and data masking modes, and makes scanning code easier.
	Data and error correction keys: These modes save actual data.
	Quiet zone: Its role is to separate itself from other surrounding content.

3. QR code feature

The data value in the QR code contains repeated information (redundant value). Therefore, even if up to 30% of the two-dimensional code structure is destroyed, the readability of the two-dimensional code will not be affected. The QR code has a storage space of up to 7089 digits or 4296 characters, including punctuation marks and special characters, which can be written into the QR code.

4. Create and recognize QR code

Install(Note: if you use jetbot-mini official image, you can skip this step)

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

- Create

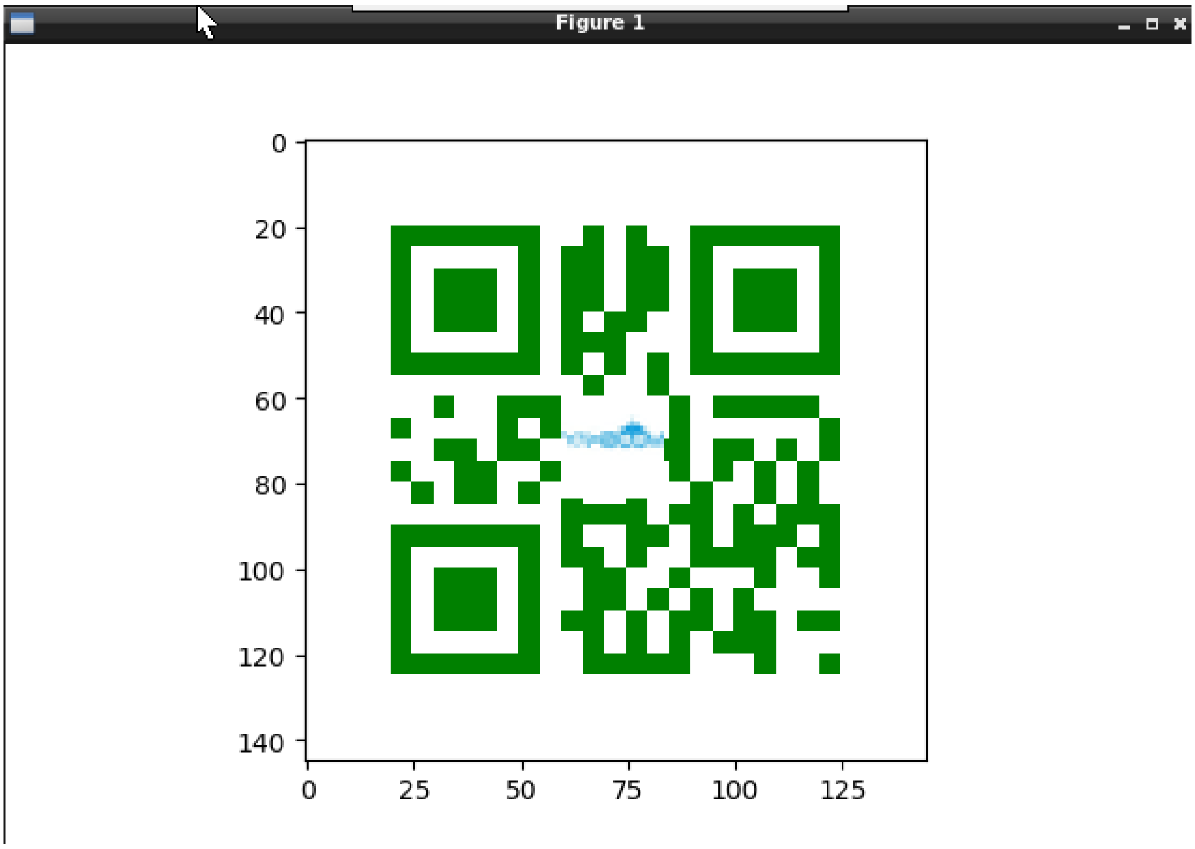
Create qrcode object

```
    '''
    Parameter:
     version: An integer ranging from 1 to 40, which controls the size of the QR
code (the minimum value is 1, which is a 12×12 matrix).
              If you want the program to determine it automatically, set the
value to None and use the fit parameter.
     error_correction: Control the error correction function of the QR code. The
following 4 constants can be used.
        ERROR_CORRECT_L: About 7% or less of errors can be corrected.
        ERROR_CORRECT_M (default): About 15% or less of errors can be corrected.
         ROR_CORRECT_H: About 30% or less 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 grids contained in the border (the distance
between the QR code and the picture border) (the default is 4, which is the
minimum value)
    '''
    qr = qrcode.QRCode(
        version=1,
        error_correction=qrcode.constants.ERROR_CORRECT_H,
        box_size=5,
        border=4,)
```

QR code to add logo

```
    # If the logo address exists, add the logo picture
    my_file = Path(logo_path)
    if my_file.is_file(): img = add_logo(img, logo_path)
```

Input ***python3 +.py execute file***, then input the content to be generated, and press 【Enter】.

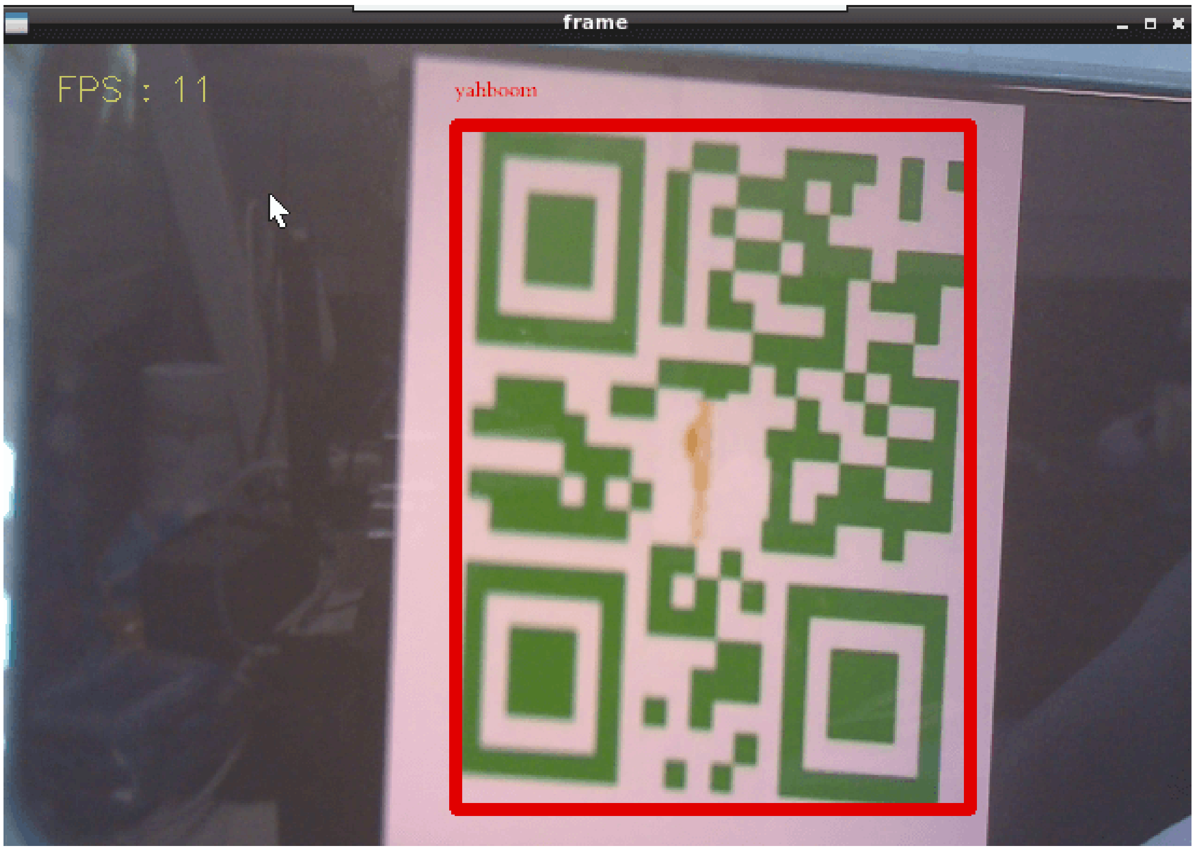


- recognize

```
def decodeDisplay(image, font_path):
    gray = cv.cvtColor(image, cv.COLOR_BGR2GRAY)
    # convert the output Chinese characters into Unicode encoding 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'
         # Convert to string
        barcodeData = barcode.data.decode(encoding)
        barcodeType = barcode.type
        # Plot the data and type on the image
        pilimg = Image.fromarray(image)
        # Create draw
        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)
        # Convert PIL image to cv2 image
        image = cv.cvtColor(np.array(pilimg), cv.COLOR_RGB2BGR)
        # Print QR code data and QR code type to the terminal
        print("[INFO] Found {} barcode: {}".format(barcodeType, barcodeData))
    return image
```

- Phenomenon show

Input ***python3 +.py execute file***, then input the content to be generated, and press 【Enter】.



Code path: ~/home/jetson/workspace/catkin_ws/src/jetbot_ros/scripts/simple_qrcode