8、QR code recognition

8.1 Introduction to gameplay

This course mainly uses the robot's camera to obtain the camera picture, calls the pyzbar library to analyze the image, and analyzes the position and information of the QR code in the image. The QR code used this time is QRCode. You can find a QR code generator online to generate a QR code as a test.

A QR code is a black and white graphic that records data symbol information and is distributed on a plane (in a two-dimensional direction) according to certain rules with certain geometric figures. It cleverly uses the "0" and "0" that form the basis of the computer's internal logic. The concept of 1" bit stream uses several geometric shapes corresponding to binary to represent text numerical information. Each code system has its own specific character set. Each character occupies a certain width and has a certain verification function.

8.2 Core content analysis

Initialize the hexapod robot and set the camera pan/tilt angle. The default is S1=90 and S2=30. The initial angle can be modified according to actual needs.

```
from MutoLib import Muto
g_bot = Muto()
g_bot.Gimbal_1_2(90, 30)
```

Import the QR code parsing library pybar

```
import pyzbar.pyzbar as pyzbar
from PIL import Image
```

If pybar is not installed on your system, please open a terminal and run the following command to install it.

```
pip3 install pyzbar
sudo apt install libzbar-dev
```

Create a new QR code detection task, which mainly converts the camera image into a grayscale image, and then passes it to the decodeDisplay function for analysis and processing.

```
def task_processing():
    global g_stop_program, g_start_function
    t_start = time.time()
    m_fps = 0
```

```
    fps = 0
    while g_camera.isOpened():
        if g_stop_program:
            break
        ret, frame = g_camera.read()

        if g_start_function:
            payload, (x, y, w, h) = detect_qrcode(frame.copy())
            if payload != None:
                cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 225, 255), 2)
                cv2.putText(frame, payload, (x, y - 10),
cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 225, 255), 1)
                print("payload:", payload)
            cv2.putText(frame, "START " + str(int(fps)), (10,20),
cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0,255,255), 1)
            # time.sleep(.010)

        m_fps = m_fps + 1
        fps = m_fps / (time.time() - t_start)
        if (time.time() - t_start) >= 2:
            m_fps = fps
            t_start = time.time() - 1
        if not g_start_function:
            cv2.putText(frame, "FPS " + str(int(fps)), (10,20),
cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0,255,255), 2)

        # 图像传输给显示组件 The image is transmitted to the display component
        image_widget.value = bgr8_to_jpeg(frame)
```

The decodeDisplay function compares the incoming images. The image number is passed into the grayscale image, and display is passed into the original image. The meaning of this is that the grayscale image recognition result can be written to the original image for display.

```
def detect_qrcode(image):
    # 转为灰度图像 Convert to grayscale image
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    barcodes = pyzbar.decode(gray)
    for barcode in barcodes:
        # 提取二维码的数据和边界框的位置 The data of the QR code and the position of
the bounding box are extracted
        (x, y, w, h) = barcode.rect
        barcodeData = barcode.data.decode("utf-8")
        barcodeType = barcode.type
        # print("[INFO] Found {} barcode: {}".format(barcodeType, barcodeData))
        return barcodeData, (x, y, w, h)
    return None, (0, 0, 0, 0)
```

Start a daemon thread, run the camera recognition task, and display the camera image.

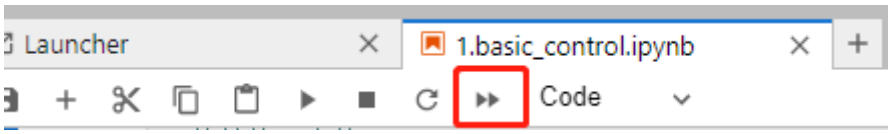
```
thread1 = threading.Thread(target=task_processing)
thread1.setDaemon(True)
thread1.start()
```

8.3 operation

Open the jupyterLab client and find the code path:

```
muto/Samples/AI_Samples/08_qrcode_recognition/qrcode_recognition.ipynb
```

Click to run all cells, and then scroll to the bottom to see the generated controls.



When the display control is started, it only displays the camera image. You need to click the Start button to turn on the QR code recognition function. The robot will recognize the QR code and display the recognized data on the QR code.

Finally click the Close_Camera button to close the camera.