Shanghai Jiaotong University

Visual Scene Real-time Analysis

Automatic QR Code Recognition

Author:
Yanhao LI
118260910036

Supervisor:
Hao LI

November 22, 2019

上海交大-巴黎高科卓越工程师学院
Ecole d'Ingénieurs SJTU-ParisTech

1 Task description

Nowadays QR code is in common use everywhere in our life. Automatic QR code recognition is currently a mature technique. Our task is to implement an algorithm to recognize a QR code and output a binary matrix. The QR code size is 42x42, thus the binary matrix is of size 42x42 with values equal to 1 for white and 0 for black.

2 Solution

To extract a regular binary matrix, we need a QR code in a regular shape, ideally a square. However, the scanned QR code is usually distorted due to an irregular scanning direction. We need to first reform the scanned QR code in a square shape, then identify the color of each small square and output 0 or 1. The solution is divided in the steps as follows:

- Read the QR code image and convert it to a gray scale image.
- Identify three boxes in the shape of 回. The shape of 回 contains three successive contours from outside to inside. We utilize findContour() function of opencv library to identify the contours that encircle a second contour, with the latter encircling a thrid contour.
- Determine the position of each box of 回. We have extracted three boxes in the previous step, but still don't know which position each box belongs to. We want to identify the top left, bottom left and top right boxes. Since the three boxes is located on three summits of a straight triangle if the QR code is a square, in most cases the side corresponding to the top left box after distorsion is still the longest side. We thus find out the longest side of the triangle, and also its corresponding top left box. Then we find out the rest of two boxes by calculating the cross product of their corresponding side vectors.
- For each box of 回, extract its four corner points using the opencv function goodFeaturesToTrack(). Then identify the positions of the four corner points according to their relative positions to the other two boxes.
- So far, we have identified 12 corner points of 3 boxes in the scanned QR code image, and we know the exact positions of these 12 poins in a regular square QR code. We use the findHomography() function to find out the homography matrix between the distorted QR code and regulare QR code.
- Reform the QR code using warpPerspective() function with the obtained homography matrix. Define a grid of size 42x42 on the square QR code.

Calculate the mean value of pixels for each small square. If the mean value is superior to 128, the small square is considered white, else considered black. Set the corresponding element of the output matrix to 1 or 0.

- Write the output matrix to a txt file.

3 Demonstration

Run the executable file QRparser with the command:

```
$ ./QRparser <input_image> <output_file>
```

For example:

```
$ ./QRparser 6_LI_Yanhao.bmp output.txt
```

Or run for debug:

```
$ DEBUG=1 ./QRparser 6_LI_Yanhao.bmp output.txt
```

You can also use the following commands to build the source code and test:

```
$ mkdir build
$ cd build
$ cmake ..
$ make
$ make test
```

4 Evaluation

This program is able to recognize a QR code in a blank background, as is given on moodle. The regularization of image using homography transform is shown below:

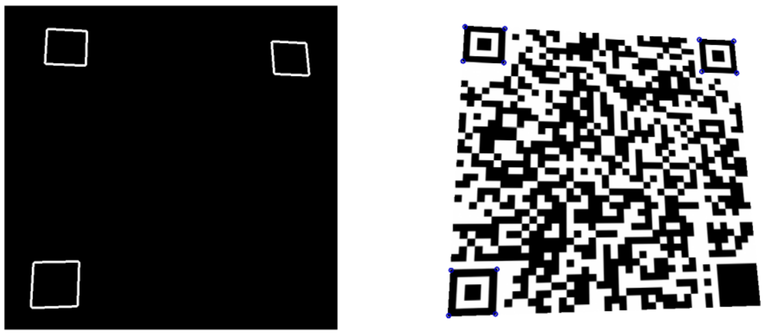

Figure 1: Extract 3 boxes and detect 12 corner points

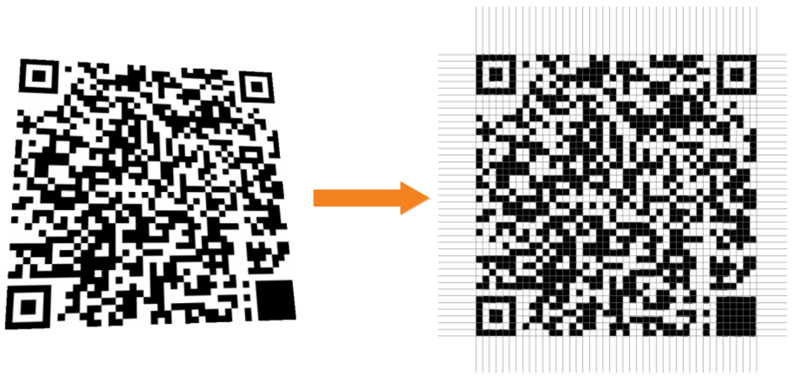

Figure 2: Homography transformation

But the program is prone to background interference, since the recognition algorithm of the shape of 回 may mistake some background objects, which needs improvement for a more robust detection.