## 2021/11/20 陳嘉政 r10922171 HW6

## 程式碼:

- 1. 先將 lena 做二值化,方法和之前相同
- 2. 二值化過後的 lena 做 downsampling: input 為二值化影像,另外宣告一個 64x64 矩陣,每個(row,col)的值為對應 input (row,col)為8 的倍數那些點的值。

3. 宣告一個 result 矩陣並全清成 0,然後將 downsampled 後的影像每個點做,若值為 255,就由此 pixel 劃出設好地矩形範圍,進入 Yokoi function

```
/*Yokoi connected number*/
Mat result = Mat::zeros(64, 64, CV_8UC1);
for (int i = 0; i < downsampled_image.rows; i++) {
    for (int j = 0; j < downsampled_image.cols; j++) {
        if (downsampled_image.at<uint8_t>(i, j) = 255) {
            int startRow = i > 0 ? 1 : 0;
            int startCol = j > 0 ? 1 : 0;
            int sizeRow = startRow = 1 ? 3 : 2;
            int sizeCol = startCol = 1 ? 3 : 2;
            sizeRow = i = 63 ? 2 : sizeRow;
            sizeCol = j = 63 ? 2 : sizeCol;
            Mat temp = downsampled_image(cv::Rect(j - startCol, i - startRow, sizeCol, sizeRow));
            result.at<uint8_t>(i, j) = Yokoi(temp, i, j);
        }
}
```

4.將每個情況列舉出來,每個情況 pixel 都去算他的四個 corner 找出 r,q,s。如果四個都是 r 回傳 5,否則計算 q 數目並回傳,將回傳數字 記錄在 result 的對應 pixel 中

```
Yokoi(Mat imput,int row, int col)
int di[] = {0, -1, 0, 1, 1, -1, -1, 1};
int dj[] = {1, 0, -1, 0, 1, 1, -1, -1};
char a[4] = { 0 };
int n;
```

### (1)左上角

### (2)右上角

## (3)左下角

# (4)右下角

# (5)上方

## (6)下方

## (7)左方

# (8)右方

# (9)正常情況

```
else {
    int i = 1, j = 1;
    for (n = -1; n < 3; n+) {
        if (input.atcuint8_to(i, j) == input.atcuint8_to(i + di[n + 1], j + dj[n + 1])) (
        if (input.atcuint8_to(i, j) == input.atcuint8_to(i + di[n + 2], j + dj[n + 2]) || input.atcuint8_to(i, j) |= input.atcuint8_to(i + di[(n + 6) % 8], j + dj[(n + 6) % 8]))
        a[n + 1] = 'q';
        else
            a[n + 1] = 'r';
        }
        else
            a[n + 1] = 's';
    }
    if (a[0] == 'r' &\data a[2] == 'r' &\data a[3] == 'r')
        return 5;
    else (
        int count = 0;
        for (n = 0; n < 4; n++)
        if (a[n] == 'q')
            count++;
        return count;
    }
}</pre>
```

# 5.利用 txt 檔輸出矩陣,並截圖調整長寬

```
fstream file("matrix.txt", ios::out);
for (int i = 0; i < result.rows; i++) {
    for (int j = 0; j < result.cols; j++) {
        if (result.at<uint8_t>(i, j) != 0)
            file << unsigned(result.at<uint8_t>(i, j));
        else
            file << " ";
    }
    file << endl;
}
file.close();</pre>
```

#### 結果:

11111111	12111111111122322221 11111111111
15555551	115555555511 2 11 11 1155555555511
15555551	115555555511 2 11 11 115555555551 21 1 2 11555512 2112221 155555555
15555551	1 2 155112 22221511 155555555511 1
15555551	22 2442 22 424 45555555511 1
15555551	22 2112 22 121 155555555511
15555551	1 2 21 2 1 1 15555555551
15111551	12 1 121111 1321 1333333333311
	1322 1155551111 1555555555551
111 1551 11 1551	1 12155555511 1555555555511
21 1551	21133333311 1331113333311
1 1551	2 155555555511 1551 115551 1
1551	2 155555555511 1551 115551 1
1551	11211333333331 1331 13311 12
1551	1 1555555555555555555551 15551 15551 15551 15551 15551 15551 1555 1555555
1551	1 22211555555555511 1151 11 1151
1551	2 22 1 10000000000000000000000000000000
	2 1 11555555555551 151 115551 115551
1551	2 115555555555111511155511 115551
1551	2 11555555555555555111511155511 115551 12 115555555555
1551	11 22155555555555555555555112 1155551
1551	111 22 1555555555555555555551 1 1555551
1551	1511 1 125112111112111555555555111 11555551
1551	15521 1 121 1 11 1 15555555111 155555551
1551	1151 132 2 1155555111 115555551
1551	12 1155555555555555555555511 155551 11 2215555555555555555555555511 1555551 111 22 15555555555555555555511 1555551 1511 1 12511211112111555555555111 1555551 15521 1 121 1 11 1 155555511 15555551 1551 132 2 11555511 121 15555551 151 322 11555511 121 15555551 1221 2 1555551 131 1155555551 1221 2 1555551 1 1155555551
1551	1221   2
1551	2 1 1155555511 1 1155555551 2 1155555551 1 1555555551 2 11555555551 211555555551 1 115555555551 15555555555
1551	2 1155555551 1 1555555551 2 11555555551 21155555551 1 115555555551 15555555551
1551	2 1155555551 21155555551
1551	1 11555555551 1555555551
1551	1 11511115555521 1 115555555551
1551	1 1 11111 115551 2 1555555551 131 1121 112
1551	131 111 15111 2 155555555551
1551	121 1121 1 111 1 2 115555555555
1551	11 111 1 221 11 1 2 1555555555555
1551	12 1 21 121 11 1111 2 155555555555
1551	1 12 22 151111111551 2 1155555555555
1551	1 12 22 151111111551 2 11555555555551 1 2 255555115511
1551	2 22 12555551 15551 1 1555555555555
1551	1 1555511 11511 2 11555555555555
1551	21 155551 1 151 2 15555555555555 2 15555112 151 2 15555555555
1551	2 15555112 151 2 15555555555555
1551	1 1 1 1155555511111 2 155555555555555
1551	2 22 111511111212 2115555555555555
1551	1 12 151 2 1 15555555111555551
1551	1 12 151 2 1 15555555111555551 11555551 11555551 11555551 11555551 11555551 1155551 115551 115551 1555551 1555551 155551 155551 155551 155551 155551 15551 1 1 1 1 155511 2 11 115511 1 1 1555555511 2111 15511 1 1 155555555
1551	1111111 15555551 1555551
1551	115551 15555551 1555511
1551	15551 21111111 155511
11521	1 12 122155511 2 11 115511
1 151	1 1 155555111 2111 15511
22 1511	1 15555555111 155111 1511
22 1511	1 1555555551 155551 1151
2 151	1 1115555555511 155511 1511
2 1521	
2 151	1 1555555555551 15551 12151 121 155555555
2 1511	15555555555551 115551 1511
21 1511	11 1555555555551 111111151
11 151	115555555555511 111511
11 151	15555555555551 151
11 151	1155555555555555 211
11 151	1155555555555511 1
11 151	15555555555551
11 111	1211111111111111
(E-0) (O)(C-1)	