# **Computer vision**

## Homework6

Yokoi Connectivity Number

## **Description**

Write a program which counts the Yokoi connectivity number on a downsampled image(lena.bmp).

#### (1) Binarize the benchmark image lena

Use the binarizing function in hw2 to obtain a binarized image of Lena.

```
□void binarize(Mat img) {
□ for (int i = 0; i < :
□ for (int j = 0; i < :
□ if (img.at<
        for (int i = 0; i < img_rows; i++) {</pre>
             for (int j = 0; j < img_cols; j++) {
                   if (img.at<uchar>(i, j) < 128) {//0~127
                        img.at<uchar>(i, j) = 0;
                  else {
                        img.at<uchar>(i, j) = 255;
```

#### (2) Downsampling Lena from 512x512 to 64x64

Create a matrix in size 64x64. For each pixel in the new matrix take the topmost-left pixel as the downsampled data.

```
pvoid downsample(Mat ori, Mat down) {
     for (int i = 0; i < 64; i++) {
         for (int j = 0; j < 64; j++) {
             down.at<uchar>(i, j) = ori.at<uchar>(i * 8, j * 8);
```

#### (3) Count the Yokoi connectivity number on a downsampled lena using 4-connected

i. Obtain the value of xo~x8.

### Corner Neighborhood (for corresponding $x_i$ )

<i>x</i> <sub>2</sub>	$x_6$
	$x_1$

_		
<i>x</i> <sub>7</sub>	$x_2$	
<i>x</i> <sub>3</sub>		

$x_3$		
<i>x</i> <sub>8</sub>	<i>x</i> <sub>4</sub>	



	$x_0$	$x_1$
5		



ii. Calculate the value of a1~a4 using the function mentioned in the lecture.

$$h(b,c,d,e) = \begin{cases} q & \text{if } b = c \text{ and } (d \neq b \ \lor e \neq b) \\ r & \text{if } b = c \text{ and } (d = b \land e = b) \\ s & \text{if } b \neq c \end{cases}$$

```
char h(int b, int c, int d, int e) 
char h(int b, int c, int d, int e) 
char h(int b, int c, int d, int e) 
char h(int b, int c, int d, int e) 
                                   return 'q';
                         else if (d == b && e == b) {
                                 return 'r';
42
                    else {
                            return 's';
```

iii. Calculate the connectivity number of the current pixel the function mentioned in the lecture and store it in the output matrix

```
f(a_1, a_2, a_3, a_4)
              5 if a_1 = a_2 = a_3 = a_4 = r
   n where n = number of \{a_k | a_k = q\}, otherwise
```

```
f(int a1, int a2, int a3, int a4) {
if (a1 == 'r' && a2 == 'r' && a3 == 'r' && a4 == 'r') {
    return 5;
else {
    int count = 0;
    if (a1 == 'q') {
        count++;
    if (a2 == 'q') {
        count++;
    if (a3 == 'q') {
        count++;
    if (a4 == 'q') {
        count++;
    return count;
```

#### Source code

```
id yokoi(Mat down) {
 for (int i = 0; i < 64; i++) {
      for (int j = 0; j < 64; j++) {
          if (down.at < uchar > (i, j) == 0) {
             matrix[i][j] = 0;
              continue;
         x[0] = down.at < uchar > (i, j);
             x[1] = down.at < uchar > (i, j + 1);
             x[3] = down.at < uchar > (i, j - 1);
          if (i != 0) {
             x[2] = down.at < uchar > (i - 1, j);
             x[4] = down.at < uchar > (i + 1, j);
          if (i != 63 && j != 63) {
             x[5] = down.at < uchar > (i + 1, j + 1);
          if (i != 0 && j != 0) {
             x[7] = down.at < uchar > (i - 1, j - 1);
          if (i != 0 && j != 63) {
             x[6] = down.at<uchar>(i - 1, j + 1);
          if (i != 63 && j != 0) {
             x[8] = down.at < uchar > (i + 1, j - 1);
         char a[4];
          a[0] = h(x[0], x[1], x[6], x[2]);
         a[1] = h(x[0], x[2], x[7], x[3]);

a[2] = h(x[0], x[3], x[8], x[4]);
         a[3] = h(x[0], x[4], x[5], x[1]);
         matrix[i][j] = f(a[0], a[1], a[2], a[3]);;
```

## <u>Result</u>

Microsoft Visual Studio 偵錯主控	台			
1 1 1 1 1 1 1 5 5 5 5 5 5 5 1 5 5 5 5 5	1 2 1 1 1 1 2 1 2 2 2 1	5 5 5 5 5 5 5 5 5 1 1 1 1 5 5 5 5 5 1 1 2 1 5 5 1 1 2 2 2 2 1 1 2 2 2 2 2 1 2	2 3 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
5 5 5 5 5 1 5 1 1 1 5 5 1 1 1 1 5 5 1 1 1 5 5 1 1 5 5 1 1 5 5 1 1 5 5 1		2 1 1 2 2 2 2 2 1 2 1 2 1 12 1 1 1 3 2 2 1 1 5 5 5 1 1 2 1 5 5 5 2 1 5 2 1 5 1 1 2 1 5 5 1 1 2 1 5 5 2 2 2 2 2 1 1 5 5	5 1 1 1 1 1 5 5 5 5 5 1 1 1 5 5 5 5 5 5	1555555555551 21 155555555555551 1 15555555555551 1 15555555555
5.5.5.5.5.5.5.5.5.5.5.5.5.5.5.5.5.5.5.	1 2 2 2 1 2 1 1	1 2 1 1 2 1 1 1 1 3 2 2 1 1 5 5 5 5 1 2 1 1 5 5 5 5 5 5 5 5	1 1 1 1 1 1 1 1 5 5 5 5 5 5 5 5 5 5 5 5	1 1 5 1     1 1     1 1 5       1 5 1     1 1 1 1 1     1 5 5       1 5 1     1 1 5 5 5 1     1 1 5 5       1 1 5 1 1 1 5 5 5 1 1     1 1 5 5 5       5 5 5 5 5 5 5 5 5 5 1     1 5 5 5       5 5 5 5 5 5 5 5 5 1 1 2     1 1 5 5 5
1 5 5 1 1 5 5 1	1 1 1 1 5 1 1 1 5 5 2 1 1 5 1 1 5 1 2 2 2 2	2 2 1 5 5 5 5 1 1 2 5 1 1 2 1 1 1 2 1 1 1 3 2 2 3 2 2 1 2 1	5 5 5 5 5 5 5 5 5 5 5 5 5 1 1 1 1 5	15   1   1   1   1   1   1   1   1   1
1 5 5 1 1 5 5 1	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	! ! ! 1		1
1551 13 1551 1 1551 2 1551 1 1551 1	i i	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	121 11 1111 151111111551 1555551115511 12555551 15551	1
1 5 5 1 1 5 5 1	1 2 1	1 1 2 2 1 2	155551 1 151 15555112 151 1155555511111 1151111212 151 2 1 11111111 115551 115551 122155511	1
1 1 5 2 1 1 5 1 2 1 5 1 1 2 1 5 1 1 2 1 5 1 1 2 1 5 2 1	1	1 2 1 1 1 1 1 1 2 1	1 1 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5	2 1 1 1 5 5 1 1
2		11	1 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5	5 5 1 1

## Reference:

1. Lecture slide (CV1\_CH6\_2020) - p.61~66