HW 6: Yokoi Connectivity Number

Source Code

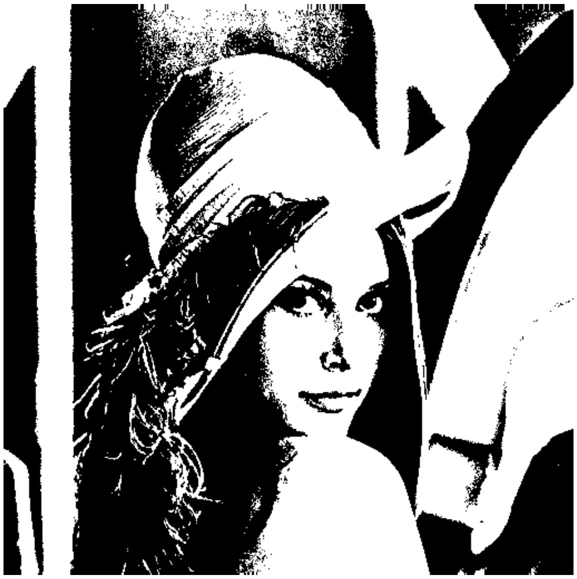
All questions are written in Python code, please refer to the file “main.py”.
All images will be stored in the folder “res” (automatically create a new folder).
In accordance with the **FAQ** of course website:

- All parts of the question are written from scratch, except for plotting images

Answer

1. Binarize
   Algorithm:
   1) Iterate grayscale image
   2) For every pixel, convert to 0 for any value less than 128 and 255 for the rest.

```
binarized = img.copy()
for i in range(height):
    for j in range(width):
        if binarized[i][j] < 128:
            binarized[i][j] = 0
        else:
            binarized[i][j] = 255
cv2.imwrite("res/binarized.bmp", binarized)
```

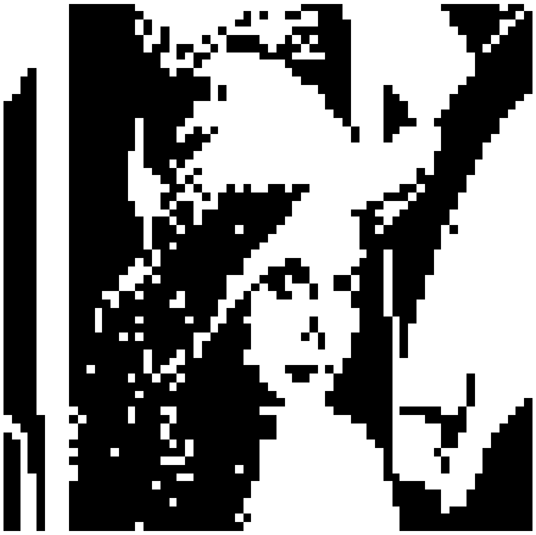


2. Downsample

Algorithm:

1) Make array of 64x64
2) Take values from binarized image with factor of 8

```
down = np.zeros(shape=(int(height/8), int(width/8)))
for i in range(down.shape[0]):
    for j in range(down.shape[1]):
        down[i][j] = binarized[i*8][j*8]
cv2.imwrite("res/downsampled.bmp", down)
```



3. Yokoi Connectivity Number

Algorithm:

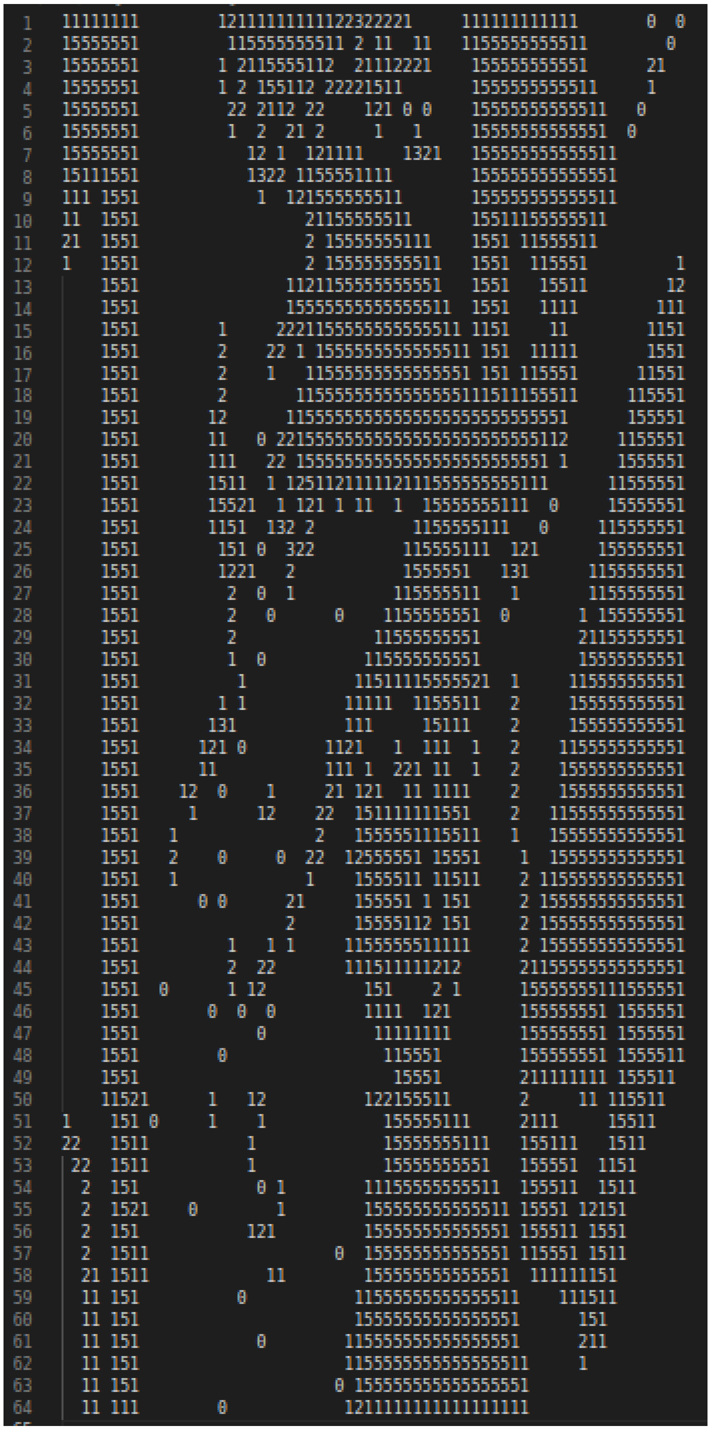
1) Iterate pixels in downsampled image
2) Find neighborhood pixels around pixel in interest
3) Count *a1, a2, a3, a4*
4) Count *f(a1, a2, a3, a4)*

```
def neighborhoodPixels(img, pos):
    pixels = np.zeros(shape=(3,3))
    col, row = pos
    for i in range(3):
        for j in range(3):
            xdest = col + i - 1
            ydest = row + j - 1
            if ((0 <= xdest < img.shape[0]) and (0 <= ydest <
img.shape[1])):
                pixels[i][j] = img[xdest][ydest]
            else:
                pixels[i][j] = 0
    return pixels
```

```
def hFunc(b, c, d, e):
    if (b == c):
        if ((d != b) or (e != b)):
            return 'q'
        elif ((d == b) and (e == b)):
            return 'r'
    else:
        return 's'

def fFunc(a1, a2, a3, a4):
    if [a1, a2, a3, a4].count('r') == 4:
        return 5
    else:
        return [a1, a2, a3, a4].count('q')

def yokoiNumber(img):
    result = np.full(img.shape, ' ')
    for i in range(img.shape[0]):
        for j in range(img.shape[1]):
            if img[i][j] != 0:
                pixels = neighborhoodPixels(img, (i,j))
                result[i][j] = fFunc(
                    hFunc(pixels[1][1], pixels[1][2], 
pixels[0][2], pixels[0][1]),
                    hFunc(pixels[1][1], pixels[0][1], 
pixels[0][0], pixels[1][0]),
                    hFunc(pixels[1][1], pixels[1][0], 
pixels[2][0], pixels[2][1]),
                    hFunc(pixels[1][1], pixels[2][1], 
pixels[2][2], pixels[1][2])
                )
    return result
```



```
11111111        12111111111122322221     111111111111       0  0
15555551         115555555511 2 11  11   1155555555511        0
15555551        1 2115555112  21112221    155555555551      21
15555551        1 2 155112 22221511       1555555555511     1
15555551         22 2112 22    121 0 0    15555555555511   0
15555551         1  2  21 2     1   1     15555555555551  0
15555551           12 1  121111    1321   155555555555511
15111551           1322 1155551111        155555555555551
111 1551            1  121555555511       155555555555511
11  1551                 21155555511      15511155555511
21  1551                 2 15555555111    1551 11555511
1   1551                 2 155555555511   1551  115551         1
    1551               1121155555555551   1551   15511        12
    1551               15555555555555511  1551   1111        111
    1551        1     2221155555555555511 1151    11        1151
    1551        2    22 1 1555555555555511 151  11111       1551
    1551        2    1   11555555555555551 151 115551      11551
    1551        2       11555555555555555111511155511     115551
    1551       12      11555555555555555555555555551      155551
    1551       11   0 221555555555555555555555555112     1155551
    1551       111   22 155555555555555555555555551 1     1555551
    1551       1511  1 125112111112111555555555111      11555551
    1551       15521  1 121 1 11  1  15555555111  0     15555551
    1551       1151  132 2          1155555111   0     115555551
    1551        151 0  322         115555111  121      155555551
    1551        1221   2           1555551   131      1155555551
    1551         2  0  1          115555511   1       1155555551
    1551         2   0      0    1155555551  0       1 155555551
    1551         2              11555555551          21155555551
    1551         1  0          115555555551          15555555551
    1551          1           11511115555521  1     115555555551
    1551        1 1          11111  1155511   2     155555555551
    1551       131           111     15111    2     155555555551
    1551      121 0        1121   1  111  1   2    1155555555551
    1551      11           111 1  221 11  1   2    1555555555551
    1551    12  0    1     21 121  11 1111    2    1555555555551
    1551     1      12    22  151111111551    2   11555555555551
    1551   1              2   1555551115511   1   15555555555551
    1551   2    0     0  22  12555551 15551    1  15555555555551
    1551   1             1    1555511 11511    2 115555555555551
    1551      0 0      21     155551 1 151     2 155555555555551
    1551               2      15555112 151     2 155555555555551
    1551         1   1 1     1155555511111     2 155555555555551
    1551         2  22       111511111212      21155555555555551
    1551  0      1 12          151    2 1      15555555111555551
    1551       0  0  0         1111  121       155555551 1555551
    1551            0           11111111       155555551 1555551
    1551        0                115551        155555551 1555511
    1551                          15551        211111111 155511
    11521      1   12          122155511       2     11 115511
1    151 0     1    1            155555111     2111     15511
22   1511          1             15555555111   155111   1511
 22  1511          1             15555555551   155551  1151
  2  151            0 1        11155555555511  155511  1511
  2  1521    0        1        155555555555511 15551 12151
  2  151           121         155555555555551 155511 1551
  2  1511                   0  155555555555551 115551 1511
  21 1511             11       155555555555551  111111151
  11 151          0           11555555555555511    111511
  11 151                      15555555555555551      151
  11 151              0      115555555555555551      211
  11 151                     1155555555555555511     1
  11 151                    0 155555555555555551
  11 111        0            1211111111111111111
```