

Computer Vision Hw6 Report

I. Implementation

Down sampling and binarization, make 512x512 into 64x64 binarized lena. First initialized an empty 64x64 2D array. Then loop each pixel and binarize it.

```
# Downsampling and binarize
downsampled_lena = [[0 for j in range(64)] for i in range(64)]

for i in range(64):
    for j in range(64):
        pixel = lena_img[i*8][j*8]
        if pixel < 128:
            downsampled_lena[i][j] = 0
        else:
            downsampled_lena[i][j] = 255
```



After that, for each pixel in 64x64 binarized lena that not equal to 0, find its neighbor for given center (x y) and find the yokoi connectivity. The yokoi connectivity is depend on H and F function according to lecture note.

```
yokoi_lena = [[' ' for j in range(64)] for i in range(64)]

for i in range(64):
    for j in range(64):
        if downsampled_lena[i][j] != 0:
            yokoi_lena[i][j] = Yokoi(getNeighbors(i, j))
```

Center point will be x_0 and its neighbors' points is according lecture note $x_1, x_2, x_3, x_4, x_5, x_6, x_7$ and x_8 . (Each point must be inside of 0 to 64)

```
def getNeighbors(x, y):
    # x7 x2 x6
    # x3 x0 x1
    # x8 x4 x5
    x7 = getPixel(x-1, y-1)
    x2 = getPixel(x, y-1)
    x6 = getPixel(x+1, y-1)

    x3 = getPixel(x-1, y)
    x0 = getPixel(x, y)
    x1 = getPixel(x+1, y)

    x8 = getPixel(x-1, y+1)
    x4 = getPixel(x, y+1)
    x5 = getPixel(x+1, y+1)
    return [x0, x1, x2, x3, x4, x5, x6, x7, x8]
```

```
def getPixel(x, y):
    if (x >= 0 and x < 64) and (y >= 0 and y < 64):
        return downsampled_lena[x][y]
    else:
        return 0
```

The yokoi function return f function of h function. According to lecture note, f_1 will be x_0, x_1, x_6 and x_2 ; f_2 will be x_0, x_2, x_7 and x_3 ; f_3 will be x_0, x_3, x_8 and x_4 ; f_4 will be x_0, x_4, x_5 and x_1 .

Corner Neighborhood
(for corresponding x_i)

	x_2	x_6
		x_1

x_7	x_2	
x_3		

x_3		
x_8	x_4	

		x_1
	x_4	x_5

```
def Yokoi(lst):
    h_lst = [h(lst[0], lst[1], lst[6], lst[2]), \
             h(lst[0], lst[2], lst[7], lst[3]), \
             h(lst[0], lst[3], lst[8], lst[4]), \
             h(lst[0], lst[4], lst[5], lst[1])]
    return f(h_lst)
```

The f function is according to lecture note equation. But since that example yokoi image didn't print 0, so I replaced 0 into space.

$$f(a_1, a_2, a_3, a_4) = \begin{cases} 5 & \text{if } a_1 = a_2 = a_3 = a_4 = r \\ n & \text{where } n = \text{numberof}\{a_k | a_k = q\}, \text{ otherwise} \end{cases}$$

```
def f(lst):
    if(lst.count('r') == 4):
        return '5'
    elif(lst.count('q') == 0):
        return ' '
    else:
        return str(lst.count('q'))
```

The h function is also according to lecture note equation.

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

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

II. Result

11111111	121111111111122322221	1111111111111
15555551	115555555511 2 11 11	1155555555511
15555551	1 2115555112 21112221	155555555551 21
15555551	1 2 155112 22221511	1555555555511 1
15555551	22 2112 22 121	15555555555511
15555551	1 2 21 2 1 1	15555555555551
15555551	12 1 121111 1321	155555555555511
15111551	1322 1155551111	155555555555551
111 1551	1 121555555511	1555555555555511
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 22211555555555511 1151 11	1151 1151
1551	2 22 1 1555555555555511 151 11111	1551 1551
1551	2 1 11555555555555551 151 115551	11551 11551
1551	2 11555555555555555111511155511	115551 115551
1551	12 115555555555555555555555555551	155551 155551
1551	11 22155555555555555555555555555112	1155551 1155551
1551	111 22 1555555555555555555555555551 1	1555551 1555551
1551	1511 1 125112111112111555555555111	11555551 11555551
1551	15521 1 121 1 11 1 15555555111	15555551 15555551
1551	1151 132 2 1155555111	11555551 15555551
1551	151 322 115555111 121	15555551 15555551
1551	1221 2 1555551 131	11555551 11555551
1551	2 1 115555511 1	11555551 11555551
1551	2 1155555551	1 155555551
1551	2 11555555551	21155555551
1551	1 11555555551	15555555551
1551	1 11511115555521 1	115555555551
1551	1 1 11111 1155511 2	155555555551
1551	131 111 15111 2	155555555551
1551	121 1121 1 111 1 2	1155555555551
1551	11 111 1 221 11 1 2	15555555555551
1551	12 1 21 121 11 1111 2	15555555555551
1551	1 12 22 151111111551 2	11555555555551
1551	1 2 1555551115511 1	15555555555551
1551	2 22 12555551 15551 1	15555555555551
1551	1 1555511 11511 2	115555555555551
1551	21 155551 1 151 2	155555555555551
1551	2 15555112 151 2	155555555555551
1551	1 1 1 1155555511111 2	1555555555555551
1551	2 22 111511111212 21155555555555551	1555555555555551
1551	1 12 151 2 1 1555555111555551	15555555511555551
1551	111 121 15555551 1555551	1555555551 1555551
1551	11111111 15555551 1555551	1555555551 1555551
1551	115551 15555551 15555511	2111111111 155511
1551	15551 211111111 155511	2 11 115511
11521	1 12 122155511 2111 15511	155111 1511
1 151	1 1 155555111 155551 1151	155511 1511
22 1511	1 15555555111 155511 1511	155511 1511
22 1511	1 15555555551 155511 1511	155511 1511
2 151	1 11155555555511 15551 12151	155511 1511
2 1521	1 15555555555551 15551 1551	155511 1511
2 151	121 155555555555551 1111111151	155511 1511
2 1511	1555555555555551 1111111151	155511 1511
21 1511	11 1555555555555551 1551	155511 1511
11 151	15555555555555551 211	155511 1511
11 151	115555555555555551 1	155511 1511
11 151	155555555555555551 1	155511 1511
11 151	155555555555555551 1	155511 1511
11 111	12111111111111111111	155511 1511