R07922141
張緣彩
10/10/2018

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 x0 and its neighbors' points is according lecture note x1, x2, x3, x4, x5, x6, x7 and x8. (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, f1 will be x0, x1, x6 and x2; f2 will be x0, x2, x7 and x3; f3 will be x0, x3, x8 and x4; f4 will be x0, x4, x5 and x1.

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.

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

```
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.

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

```
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