Computer Vision HW7_R09922093 楊子萱

In this homework, I use python3 and import cv2 to read and write images and use cv2 function to translate image into the binary image. To run my code, use command line and enter python3 hw7.py (http://hw7.py) .

(a) Downsampling :

downsample lena.bmp from 512x512 to 64x64 and binarize the image.

```
def downsample(image) :
    d_row = image.shape[0]//8
    d_column = image.shape[1]//8
    for i in range (d_row):
        for j in range (d_column):
            down_sampling[i][j] = image[8 * i][8 * j]
            if down_sampling[i][j] < 128:
                down_sampling[i][j] = 0
            else :
                down_sampling[i][j] = 255
    return d_row,d_column, down_sampling
```

(b) Pair Relationship Operator

Use the Yokoi connectivity done by HW6 to generate the image used in pair relationship operator.

In the pair operator, I used the H function referring to homweork pdf :

H function: (m="1", means "edge" in Yokoi)

- $h(a,m) = \begin{cases} 1, if\ a = m \\ 0, otherwise \end{cases}$

```
def pair_relation(i, j):
    if i+1 < down_row:
        if yokoi_result[i+1][j] == 1:
            return 1
    if i-1 >= 0:
        if yokoi_result[i-1][j] == 1:
            return 1
    if j+1 < down_column:
        if yokoi_result[i][j+1] == 1:
            return 1
    if j-1 >= 0:
        if yokoi_result[i][j-1] == 1:
            return 1
    return 0
```

Output:

- $y = \begin{cases} q, if\ \sum_{n=1}^{4} h(x_n, m) < 1\ or\ x_0 \neq m \\ p, if\ \sum_{n=1}^{4} h(x_n, m) \geq 1\ and\ x_0 = m \end{cases}$

Here, I used 1 and -1 to define p and q.

```
def thining_mark():
    for i in range (down_row):
        for j in range (down_column):
            if yokoi_result[i][j] == 0:
                thining[i][j] = 0
            elif yokoi_result[i][j] == 1 and pair_relation(i,j) == 1:
                thining[i][j] = 1
            else:
                thining[i][j] = -1
```

(c) Connected Shrink Operator

I defined the function h_shrink and f_shrink referring to the homweork pdf as :

H function: (yokoi corner => "q")

- $h(b, c, d, e) = \begin{cases} 1, if\ b = c\ and\ (d \neq b\ or\ e \neq b) \\ 0, otherwise \end{cases}$

Output:

- $f(a_1, a_2, a_3, a_4, x) = \begin{cases} g, if\ exactly\ one\ of\ a_n = 1, n = 1{\sim}4 \\ x, otherwise \end{cases}$

h_shrink

```
def h_shrink(i,j,p1,p2,p3):
    v_0 = down_sampling[i][j]
    v_1 = getvalue(i,j,p1)
    v_2 = getvalue(i,j,p2)
    v_3 = getvalue(i,j,p3)
    if v_0 == v_1:
        if v_0 != v_2 or v_0 != v_3:
            return 1
    return 0
```

ps. getvalue function defined in the HW6

f_shink

```
def f_shrink(i, j, a,b,c,d):
    if a+b+c+d == 1:
        return 0
    else:
        return down_sampling[i][j]
```

ps. down_sampling is the image got from the result of downsample function.
(d) Iteration 7 times and get the result.
Every time I do the iteration, I renewed the yokoi_connectivity result and mark result to all 0 again. After the yokoi and mark, I used the connected shrink on the down_sampling image to get the result.

```
yokoi_result = np.full((down_row,down_column),0)
thining = np.full((down_row, down_column),0)
for time in range(7):
    yokoi_result = np.full((down_row,down_column),0)
    yokoi_connectivity(time)
    thining = np.full((down_row, down_column),0)
    thining_mark()
    for i in range (down_row):
        for j in range (down_column):
            if thining[i][j] == 1:
                a = h_shrink(i,j,1,6,2)
                b = h_shrink(i,j,2,7,3)
                c = h_shrink(i,j,3,8,4)
                d = h_shrink(i,j,4,5,1)
                down_sampling[i][j] = f_shrink(i, j, a, b, c, d)
cv2.imwrite("thining_result.bmp", down_sampling)
```

Result:

Downsampling	**Thining**