Computer Vision I (922 U0610) - Homework 6

Author: alanhc

ID: r10944007

Date: 10/27

README

0. create env: `conda env create -f environment.yml`
1. enter env: `conda activate ntu-cv`
2. run jupyter `jupyter notebook`

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

- Downsampling Lena from 512x512 to 64x64:
  - Binarize the benchmark image lena as in HW2, then using 8x8 blocks as a unit, take the topmost-left pixel as the downsampled data.
- Count the Yokoi connectivity number on a downsampled lena using 4-connected.
- Result of this assignment is a 64x64 matrix.
- You can use any programing language to implement homework, however, you'll get zero point if you just call existing library.

In [1]:
```
from PIL import Image
import numpy as np

# Todo: 讀檔，確定影像大小
img = Image.open("input/lena.bmp")
img = np.array(img)

h, w = img.shape

print("image shape:", img.shape)
show = Image.fromarray(img).resize((256,256))
show
```

```
image shape: (512, 512)
```

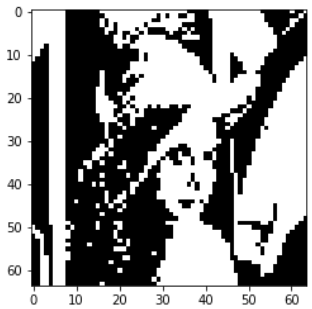
Out[1]:

In [2]:
```
# Todo: sampling
## Hint: topmost-left: 左上角點可用//8求
# Algorithm:
## 1. 根據hint，先算出一個row/col有幾個sampling點，再用*8來還原sampling點在原本的位置
## 2. >128 來產生threshold 128的影像

## 512->64.  512/64=8
# 1.
ans = np.zeros((h//8,w//8))
for y in range(h//8):
    for x in range(w//8):
        # 1.
        if (img[y*8][x*8]>=128):# 2.
            ans[y][x] = 255
```

In [3]:
```
import matplotlib.pyplot as plt
print(ans.shape)
plt.imshow(ans, cmap="gray")
img_downSampling = ans
```

```
(64, 64)
```



For 4-connectivity, the function h of four arguments is defined by

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

The function f of four arguments is defined by

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

In [4]:
```
"""
index pixels
7 2 6
3 0 1
8 4 5
"""
# Todo: Yokoi connectivity numbe
# Algorithm:
## 1. 計算 h(a,b,c,d)，*程式為f_h* ，藉由三個coner點判斷類型 q,r,s，其中b,c,d,e為帶入的kernel點xi
## 2. 計算 connectivity operator f(a1,a2,a3,a4)，其中a1,a2,a3,a4是由 h(x0,x1,x6,x2), h(x0,x2,x7,x3),
##                                                                    h(x0,x3,x8,x4), h(x0,x4,x5,x1)
## 3. 根據課本式子參考上圖，計算5或n

s_h, s_w = img_downSampling.shape

def index_values(img, y, x, n):
    # shift is a convert table reelated to index  textbook:x0,x1,x2...
    shift = {
        7:[-1,-1],
        2:[-1,0],
        6:[-1,1],
        3:[0,-1],
        0:[0,0],
        1:[0,1],
        8:[1,-1],
        4:[1,0],
        5:[1,1]
    }
    now_y = y+shift[n][0]
    now_x = x+shift[n][1]

    if (now_y>=0 and now_x>=0 and now_y<s_h and now_x<s_w):
        return img[ now_y ][ now_x ]
    else:
        return 0

# 1.
def f_h(img, pos,b,c,d,e):
    y, x = pos

    b = index_values(img, y, x, b)
    c = index_values(img, y, x, c)
    d = index_values(img, y, x, d)
    e = index_values(img, y, x, e)

    if (b==c and (d!=b or e!=b)):
        return "q"
    if (b==c and (d==b and e==b)):
        return "r"
    if (b!=c):
        return "s"
    else:
        print("=", b,c,d,e)

def f(a1,a2,a3,a4):
    if (a1==a2 and a2==a3 and a3==a4 and a4=="r"):
        return 5
    s = str(a1+a2+a3+a4)
    ## 找{a1,a2,a3,a4}有幾個q
    ct=0
    for i in range(len(s)):
        if s[i]=="q":
            ct+=1
    return ct
```

In [5]:
```
src = img_downSampling
ans = np.zeros((s_h, s_w))
for y in range(s_h):
    for x in range(s_w):
        ## 2.
        a1 = f_h(src, (y,x),0,1,6,2)
        a2 = f_h(src, (y,x),0,2,7,3)
        a3 = f_h(src, (y,x),0,3,8,4)
        a4 = f_h(src, (y,x),0,4,5,1)
        if (src[y][x]==0):
            continue
        ## 3.
        ans[y][x] = f(a1,a2,a3,a4)
ans.shape
```

Out[5]:
```
(64, 64)
```

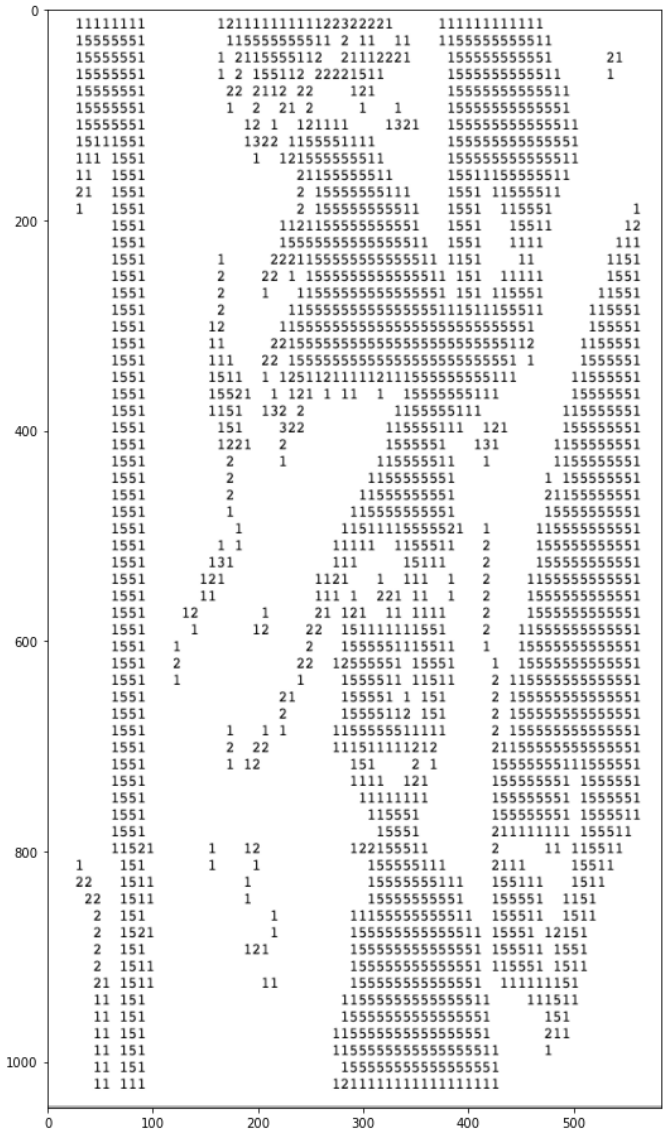
In [6]:
```
for y in range(ans.shape[0]):
    for x in range(ans.shape[1]):
        now = int(ans[y][x])
        if (now==0):
            print(" ", end="")
        else:
            print("%1d"%now, end="")
    print()
```

```
11111111        12111111111122322221     111111111111
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        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     22211555555555555511 1151    11         1151
    1551        2    22 1 1555555555555511 151  11111        1551
    1551        2    1   115555555555555 151 115551       11551
    1551        2       1155555555555555111511155511     115551
    1551       12      11555555555555555555555555551      155551
    1551       11     2215555555555555555555555555112     1155551
    1551       111   22 155555555555555555555555551 1     1555551
    1551       1511  1 125112111112111555555555111       11555551
    1551       15521  1 121 1 11  1  15555555111         15555551
    1551       1151  132 2           1155555111          115555551
    1551        151    322          115555111  121       155555551
    1551        1221   2            1555551   131       1155555551
    1551         2     1           115555511   1        1155555551
    1551         2                1155555551           1 155555551
    1551         2               11555555551           21155555551
    1551         1              115555555551           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    1555555555551
    1551    12        1     21 121  11 1111    2    1555555555551
    1551     1       12    22  151111111551    2   11555555555551
    1551   1               2   1555551115511   1   15555555555551
    1551   2              22  12555551 15551    1  15555555555551
    1551   1              1    1555511 11511    2 115555555555551
    1551                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         1 12          151    2 1      15555555111555551
    1551                       1111  121       155555551 1555551
    1551                        11111111       155555551 1555551
    1551                          115551       155555551 1555511
    1551                           15551       211111111 155511
    11521      1   12           122155511       2     11 115511
1    151       1    1             155555111     2111     15511
22   1511          1              15555555111   155111   1511
 22  1511          1              15555555551   155551  1151
  2  151              1         11155555555511  155511  1511
  2  1521             1         155555555555511 15551 12151
  2  151           121          155555555555551 155511 1551
  2  1511                       155555555555551 115551 1511
  21 1511            11         155555555555551  111111151
  11 151                       11555555555555511    111511
  11 151                       15555555555555551      151
  11 151                      115555555555555551      211
  11 151                      1155555555555555511     1
  11 151                       155555555555555551
  11 111                      1211111111111111111
```

In [7]:
```
import pandas as pd
pd.DataFrame(ans).to_csv("output/result.csv")
```

In [8]:
```
import matplotlib.pyplot as plt
img = Image.open("output/Yokoi_connectivity_number.png")
plt.figure(figsize=(16,16))
plt.imshow(img)
```

Out[8]: <matplotlib.image.AxesImage at 0x7f818b8aa340>



Ref

- https://en.wikipedia.org/wiki/Neighborhood_operation (https://en.wikipedia.org/wiki/Neighborhood_operation)
- textbook