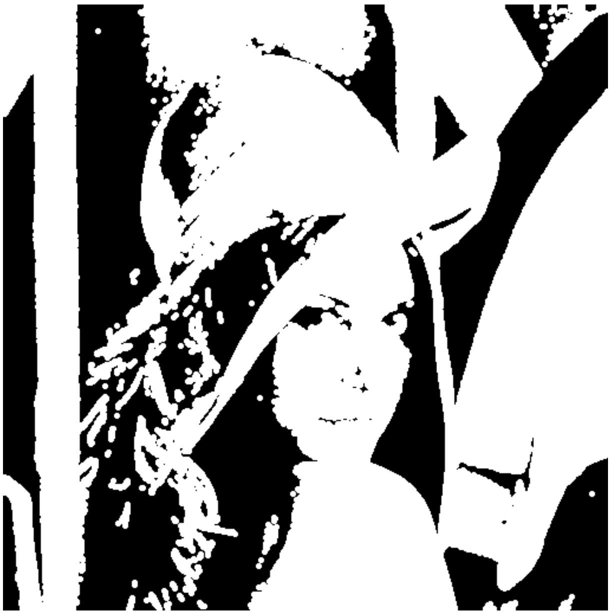
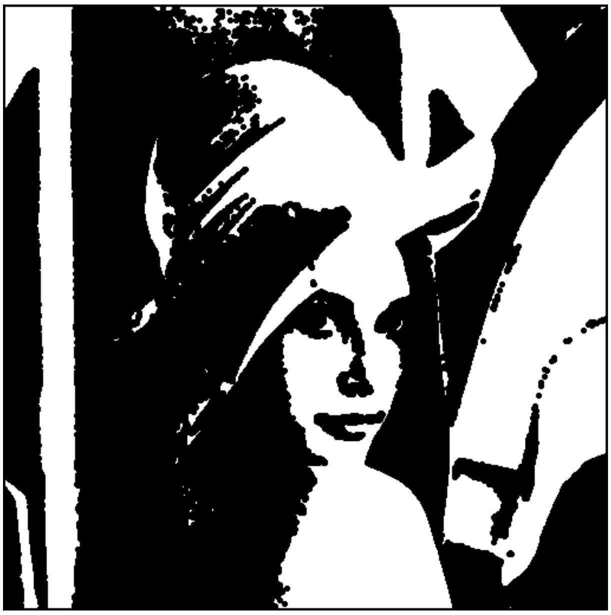
電腦視覺 CV Home Work 4 資研一 R07922003 劉濬慶

- Dilation



掃過 binary 圖上每個 pixel，若 pixel=255，設此 pixel 的點為 a，則以 a 為原點，與 kernel 的原點疊，掃過 kernel 上所有點，將值為 1 的點，對應到輸出新的全黑圖上的那點 pixel 值都設為 255

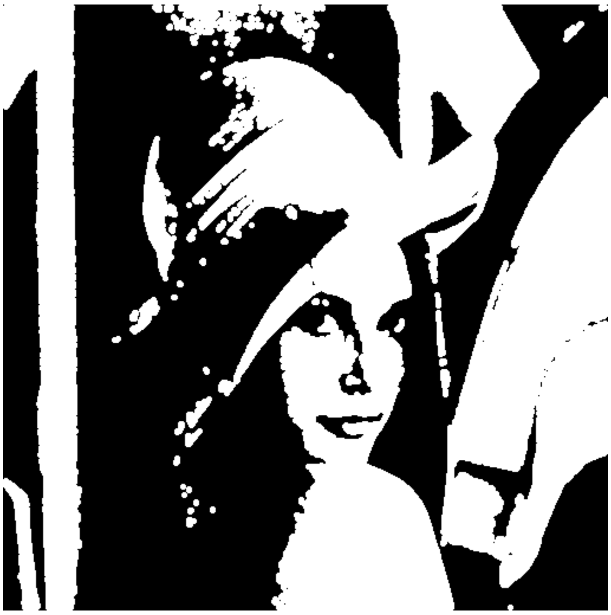
- Erosion



掃過 binary 圖上每個 pixel(不需要判斷 pixel 是否=255)，設此 pixel 的

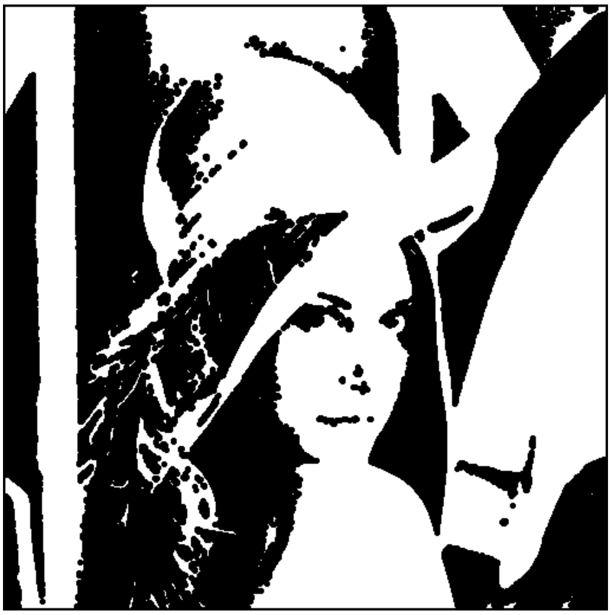
點為 a，則以 a 為原點，與 kernel 的原點重疊，掃過 kernel 上所有點，若 kernel=1 的點，在 binary 圖的位置中 ”全部都= ” 255，才將新全黑圖上 a 的 pixel 設為 255，若 kernel 為 1，但 binary 圖上的 pixel 為 0，則新全黑圖上 a 的 pixel 為 0

- Opening



先 Erosion，再 Dilation

- Closing



先 Dilation，再 Erosion

- Hit and Miss

```
from PIL import Image, ImageDraw
import matplotlib.pyplot as plt
import numpy as np

def binary_image(coulmn,row,pix,lena):
        for i in range(coulmn):
                for j in range(row):
                        if pix[i,j] < 128:
                                lena.putpixel((i,j),0)
                        else:
                                lena.putpixel((i,j),255)
        #lena.save('binary.bmp')
        return lena
def binary_component(coulmn,row,pix,lena):
        img_new=Image.new(binary.mode, binary.size)
```

```
    for i in range(coulmn):
        for j in range(row):
            if pix[i,j] < 128:
                img_new.putpixel((i,j),255)
            else:
                img_new.putpixel((i,j),0)
    return img_new
def dilation(binary,kernal,coulmn,row):
    pix=binary.load()
    img_new=Image.new(binary.mode, binary.size)
    for i in range(coulmn):
        for j in range(row):
            if pix[i,j] == 255:
                for x in range(-2,3,1):
                    for y in range(-2,3,1):
                        if kernal[x+2][y+2] == 1:
                            if (i+x >=0) and (j+y >= 0) and (i+x <
coulmn) and (j+y < row):
                                img_new.putpixel((i+x,j+y),255)
    #img_new.save('dilation.bmp')
    return img_new
def erosion(binary,kernal,coulmn,row):
    pix=binary.load()
    img_new=Image.new(binary.mode, binary.size)
    for i in range(coulmn):
        for j in range(row):
            draw=1
            for x in range(-2,3,1):
                for y in range(-2,3,1):
                    if kernal[x+2][y+2] == 1:
                        if (i+x < coulmn) and (j+y < row) and (i+x >=
0) and (j+y >= 0):
                            if pix[i+x,j+y]==0:
                                draw=0
                        else:
                            draw=0
            if draw == 1:
                img_new.putpixel((i,j),255)
    #img_new.save('erosion.bmp')
    return img_new
def opening(binary,kernal,coulmn,row):
    p=erosion(binary,kernal,coulmn,row)
    op=dilation(p,kernal,coulmn,row)
    #op.save('opening.bmp')
    return op
def closing(binary,kernal,coulmn,row):
    p=dilation(binary,kernal,coulmn,row)
    close=erosion(p,kernal,coulmn,row)
    #close.save('closing.bmp')
    return close

def hit_miss(binary,kernal_j,kernal_k,coulmn,row):
    pix=binary.load()
    component=binary_component(coulmn,row,pix,binary)
    J=erosion(binary,kernal_k,coulmn,row)
    K=erosion(component,kernal_j,coulmn,row)
    img_new=Image.new(binary.mode, binary.size)
    J_pixel=J.load()
    K_pixel=K.load()
    for i in range(coulmn):
        for j in range(row):
            if J_pixel[i,j]==255 and K_pixel[i,j]==255:
                    img_new.putpixel((i,j),255)
    #img_new.save('hit_miss.bmp')
    return img_new

lena= Image.open("lena.bmp")
pix=lena.load()
coulmn,row=lena.size
binary=binary_image(coulmn,row,pix,lena)
kernal_array=np.array([[0,1,1,1,0],
            [1,1,1,1,1],
            [1,1,1,1,1],
            [1,1,1,1,1],
            [0,1,1,1,0]])
```

```
kernal_j=np.array([[0,0,0,0,0],
                        [0,0,0,0,0],
                        [0,1,1,0,0],
                        [0,0,1,0,0],
                        [0,0,0,0,0]])

kernal_k=np.array([[0,0,0,0,0],
                        [0,0,1,1,0],
                        [0,0,0,1,0],
                        [0,0,0,0,0],
                        [0,0,0,0,0]]
                     )
#dilation(binary,kernal_array,coulmn,row)
#erosion(binary,kernal_array,coulmn,row)
#closing(binary,kernal_array,coulmn,row)
#opening(binary,kernal_array,coulmn,row)
#hit_miss(binary,kernal_j,kernal_k,coulmn,row)
```