CV Home Work 9 資研一 R07922003 劉濬慶

將原圖分成兩種不同的 convolution 以及處理方法:

1. 對兩個 mask convolution 後的值平方和開根號

   Robert、Prewitt、Sobel、Frei and Chen 用此法

2. 二是取所有 convolution 後的值得最大值

   Kirsch、Robinson、Nevatia Badu 用此法

之後再根據各自的 Threshold 去做 Binary

- robert = Robert(lena,**12**)
- prewitt = Prewitt(lena,**24**)
- sobel = Sobel(lena,**38**)
- frei_chen = Frei_Chen(lena,**30**)
- kirsch = Kirsch(lena,**135**)
- robinson = Robinson(lena,**60**)
- Nevatia_Babu(lena,**12500**)

Robert.bmp

Prewitt.bmp

Sobel.bmp

Frei_Chen.bmp

Kirsch.bmp

Robinson.bmp

Nevatia_Babu.bmp

Code

```
from PIL import Image, ImageDraw
import numpy as np


def Robert(img,threshold):
    pixel = img.load()
    img_new = Image.new(img.mode,img.size)
    array = np.zeros((img.width,img.height))
    temp1 = 0
    temp2 = 0
    mask1 = np.array([[1,0],
                      [0,-1]])
    mask2 = np.array([[0,1],
                      [-1,0]])
    for i in range(img.width):
        for j in range(img.height):
            try:
                temp1 = (pixel[i,j]*mask1[0][0] +
pixel[i+1,j+1]*mask1[1][1])**2
```

```
                        temp2 = (pixel[i,j+1]*mask2[0][1] +
pixel[i+1,j]*mask2[1][0])**2
                        array[i][j] = (temp1+temp2)**0.5
                        #img_new.putpixel((i,j),
(temp1+temp2)**0.5 )
                    except:
                        array[i,j] = pixel[i,j]
                        #img_new.putpixel((i,j),pixel[i,j])
        for i in range(img.width):
            for j in range(img.height):
                if array[i,j] < threshold:
                    img_new.putpixel((i,j),255)
                else:
                    img_new.putpixel((i,j),0)
        img_new.save("Robert.bmp")

        return img_new

def Prewitt(img,threshold):
```

```
        pixel = img.load()
        img_new = Image.new(img.mode,img.size)
        array = np.zeros((img.width,img.height))
        mask1 = np.array([[-1,-1,-1],
                          [ 0, 0, 0],
                          [ 1, 1, 1]])
        mask2 = np.array([[-1, 0, 1],
                          [-1, 0, 1],
                          [-1, 0, 1]])
        for i in range(1,img.width-1):
            for j in range(1,img.height-1):
                temp1 = 0
                temp2 = 0
                for x in range(-1,2):
                    for y in range(-1,2):
                        temp1 +=
pixel[i+x,j+y]*mask1[x+1][y+1]
                        temp2 +=
pixel[i+x,j+y]*mask2[x+1][y+1]
```

```
                array[i][j] = (temp1**2+temp2**2)**0.5
        for i in range(img.width):
            for j in range(img.height):
                if array[i,j] < threshold:
                    img_new.putpixel((i,j),255)
                else:
                    img_new.putpixel((i,j),0)
        img_new.save("Prewitt.bmp")

        return img_new

def Sobel(img,threshold):
        pixel = img.load()
        img_new = Image.new(img.mode,img.size)
        array = np.zeros((img.width,img.height))
        mask1 = np.array([[-1,-2,-1],
                          [ 0, 0, 0],
                          [ 1, 2, 1]])
        mask2 = np.array([[-1, 0, 1],
```

```
                    [-2, 0, 2],
                    [-1, 0, 1]])
    for i in range(1,img.width-1):
        for j in range(1,img.height-1):
            temp1 = 0
            temp2 = 0
            for x in range(-1,2):
                for y in range(-1,2):
                    temp1 +=
pixel[i+x,j+y]*mask1[x+1][y+1]
                    temp2 +=
pixel[i+x,j+y]*mask2[x+1][y+1]
            array[i][j] = (temp1**2+temp2**2)**0.5
    for i in range(img.width):
        for j in range(img.height):
            if array[i,j] < threshold:
                img_new.putpixel((i,j),255)
            else:
                img_new.putpixel((i,j),0)
```

```
        img_new.save("Sobel.bmp")


        return img_new


def Frei_Chen(img,threshold):
        pixel = img.load()
        img_new = Image.new(img.mode,img.size)
        array = np.zeros((img.width,img.height))
        mask1 = np.array([[-1,-(2**0.5),-1],
                                        [ 0, 0, 0],
                                        [ 1, 2**0.5, 1]])
        mask2 = np.array([[-1, 0, 1],
                                        [-(2**0.5), 0, 2**0.5],
                                        [-1, 0, 1]])
        for i in range(1,img.width-1):
                for j in range(1,img.height-1):
                        temp1 = 0
                        temp2 = 0
                        for x in range(-1,2):
```

```
                        for y in range(-1,2):
                            temp1 +=
pixel[i+x,j+y]*mask1[x+1][y+1]
                            temp2 +=
pixel[i+x,j+y]*mask2[x+1][y+1]
                    array[i][j] = (temp1**2+temp2**2)**0.5
        for i in range(img.width):
            for j in range(img.height):
                if array[i,j] < threshold:
                    img_new.putpixel((i,j),255)
                else:
                    img_new.putpixel((i,j),0)
        img_new.save("Frei_Chen.bmp")

        return img_new

def Kirsch(img,threshold):
        pixel = img.load()
        img_new = Image.new(img.mode,img.size)
```

```
array = np.zeros((img.width,img.height))
mask1 = np.array([[-3,-3, 5],
                  [-3, 0, 5],
                  [-3,-3, 5]])
mask2 = np.array([[-3, 5, 5],
                  [-3, 0, 5],
                  [-3,-3,-3]])
mask3 = np.array([[ 5, 5, 5],
                  [-3, 0,-3],
                  [-3,-3,-3]])
mask4 = np.array([[ 5, 5,-3],
                  [ 5, 0,-3],
                  [-3,-3,-3]])
mask5 = np.array([[ 5,-3,-3],
                  [ 5, 0,-3],
                  [ 5,-3, -3]])
mask6 = np.array([[-3,-3,-3],
                  [ 5, 0,-3],
                  [ 5, 5,-3]])
```

```
        mask7 = np.array([[-3,-3,-3],
                          [-3, 0,-3],
                          [ 5, 5, 5]])
        mask8 = np.array([[-3,-3,-3],
                          [-3, 0, 5],
                          [-3, 5, 5]])
        mask_list = [mask1, mask2, mask3, mask4, mask5, mask6,
mask7, mask8]
        for i in range(1,img.width-1):
            for j in range(1,img.height-1):
                temp = np.zeros(8)
                for k in range(8):
                    for x in range(-1,2):
                        for y in range(-1,2):
                            temp[k] +=
pixel[i+x,j+y]*mask_list[k][x+1][y+1]
                array[i][j] = max(temp)
        for i in range(img.width):
            for j in range(img.height):
```

```
                if array[i,j] < threshold:
                        img_new.putpixel((i,j),255)
                else:
                        img_new.putpixel((i,j),0)
        img_new.save("Kirsch.bmp")


        return img_new


def Robinson(img,threshold):
        pixel = img.load()
        img_new = Image.new(img.mode,img.size)
        array = np.zeros((img.width,img.height))
        mask1 = np.array([[-1, 0, 1],
                                [-2, 0, 2],
                                [-1, 0, 1]])
        mask2 = np.array([[ 0, 1, 2],
                                [-1, 0, 1],
                                [-2,-1, 0]])
        mask3 = np.array([[ 1, 2, 1],
```

```
                                    [ 0, 0, 0],
                                    [-1,-2,-1]])
        mask4 = np.array([[ 2, 1, 0],
                                    [ 1, 0,-1],
                                    [ 0,-1,-2]])
        mask5 = np.array([[ 1, 0,-1],
                                    [ 2, 0,-2],
                                    [ 1, 0,-1]])
        mask6 = np.array([[ 0,-1,-2],
                                    [ 1, 0,-1],
                                    [ 2, 1, 0]])
        mask7 = np.array([[-1,-2,-1],
                                    [ 0, 0, 0],
                                    [ 1, 2, 1]])
        mask8 = np.array([[-2,-1, 0],
                                    [-1, 0, 1],
                                    [ 0, 1, 2]])
        mask_list = [mask1, mask2, mask3, mask4, mask5, mask6,
mask7, mask8]
```

```
        for i in range(1,img.width-1):
                for j in range(1,img.height-1):
                        temp = np.zeros(8)
                        for k in range(8):
                                for x in range(-1,2):
                                        for y in range(-1,2):
                                                temp[k] +=
pixel[i+x,j+y]*mask_list[k][x+1][y+1]
                        array[i][j] = max(temp)
        for i in range(img.width):
                for j in range(img.height):
                        if array[i,j] < threshold:
                                img_new.putpixel((i,j),255)
                        else:
                                img_new.putpixel((i,j),0)
        img_new.save("Robinson.bmp")

        return img_new
```

```
def Nevatia_Babu(img,threshold):
        pixel = img.load()
        img_new = Image.new(img.mode,img.size)
        array = np.zeros((img.width,img.height))
        mask1 = np.array([[ 100, 100, 100, 100, 100], # 0 degree
                                  [ 100, 100, 100, 100, 100],
                                  [      0,      0,      0,      0,      0],
                                  [-100,-100,-100,-100,-100],
                                  [-100,-100,-100,-100,-100]])
        mask2 = np.array([[ 100, 100, 100, 100, 100], # 30 degree
                                  [ 100, 100, 100,    78, -32],
                                  [ 100,    92,      0, -92,-100],
                                  [    32, -78,-100,-100,-100],
                                  [-100,-100,-100,-100,-100]])
        mask3 = np.array([[ 100, 100, 100,    32,-100], # 60 degree
                                  [ 100, 100,    92, -78,-100],
                                  [ 100, 100,      0,-100,-100],
                                  [ 100,    78, -92,-100,-100],
                                  [ 100, -32,-100,-100,-100]])
```

```
mask4 = np.array([[-100,-100, 0, 100, 100], # -90 degree
                  [-100,-100, 0, 100, 100],
                  [-100,-100, 0, 100, 100],
                  [-100,-100, 0, 100, 100],
                  [-100,-100, 0, 100, 100]])
mask5 = np.array([[-100,    32, 100, 100, 100], # -60 degree
                  [-100, -78,    92, 100, 100],
                  [-100,-100,      0, 100, 100],
                  [-100,-100, -92,    78, 100],
                  [-100,-100,-100, -32, 100]])
mask6 = np.array([[ 100, 100, 100, 100, 100], # -30 degree
                  [ -32,    78, 100, 100, 100],
                  [-100, -92,      0,    92, 100],
                  [-100,-100,-100, -78,    32],
                  [-100,-100,-100,-100,-100]])
mask_list = [mask1, mask2, mask3, mask4, mask5, mask6]
for i in range(2,img.width-2):
    for j in range(2,img.height-2):
        temp = np.zeros(6)
```

```
                    for k in range(len(mask_list)):
                        for x in range(-2,3):
                            for y in range(-2,3):
                                temp[k] +=
pixel[i+x,j+y]*mask_list[k][-x+2,-y+2]
                    array[i][j] = max(temp)
        for i in range(img.width):
            for j in range(img.height):
                if array[i,j] < threshold:
                    img_new.putpixel((i,j),255)
                else:
                    img_new.putpixel((i,j),0)
        img_new.save("Nevatia_Babu.bmp")

        return img_new


lena = Image.open("lena.bmp")
robert = Robert(lena,12)
```

```
prewitt = Prewitt(lena,24)

sobel = Sobel(lena,38)

frei_chen = Frei_Chen(lena,30)

kirsch = Kirsch(lena,135)

robinson = Robinson(lena,60)

nevatia_babu=Nevatia_Babu(lena,12500)
```