Computer Vision I \_2018

Homework assignment #4

R07522717機械所製造組碩一 林温雅

Part1（此次作業僅one part）

Description:

Write programs which do binary morphological dilation, erosion, opening, closing, and hit-and-miss transform on a binary image.

Algorithm:

1. Dilation:

獲得kernel與輸入image的行列數以後，製作一個『外擴』圖檔，為的是處理kernel在尋訪時超出邊界的情況（以image為512\*512, kernel為5\*5來說，暫存圖檔為516\*516，暫存圖檔為往上、往下、往左、往右分別外擴兩列/行，外擴新增的pixel值為0，中間則就是原本輸入圖檔的值）。

接著再進行邏輯運算，若kernel中心覆蓋某p點時，kernel與image有and計算為true的情況，則將p點設為白色。

1. Erosion:

獲得kernel與輸入image的行列數以後，製作一個『外擴』圖檔，為的是處理kernel在尋訪時超出邊界的情況（以image為512\*512, kernel為5\*5來說，暫存圖檔為516\*516，暫存圖檔為往上、往下、往左、往右分別外擴兩列/行，外擴新增的pixel值為1，中間則就是原本輸入圖檔的值。特別注意這邊外擴pixel的值為1，而在dilation function內是0。ppt內之erosion照片應是忽略此點，才造成邊界都是一圈黑色）。

接著再進行邏輯運算，若kernel中心覆蓋某p點時，kernel與image之and計算結果和kernel本身相等（及整個kernel的1值都在image為白色處），則將p點設為白色。

1. Opening:

對影像進行Erosion後再Dilation

1. Closing:

對影像進行Dilation後再Erosion

1. Hit-and-Miss:

先對binary image製作黑白互補圖的互補圖，接著製作此原image與kernel\_j的erosion圖&互補圖與kernel\_k的erosion圖。最後輸出此兩張erosion結果圖重疊的部分。

Parameters:

1. In function “dilation”:

img\_rows, img\_columns #輸入圖檔的行列數

ker\_rows, ker\_columns #kernel的行列數

row\_dist, column\_dist #計算kernel中心距離邊界有多遠，主要目的是看原始圖檔要擴大多少

temp\_img #原始圖檔擴大後的暫存圖檔

new\_img #新圖檔準備接受dilation後的圖

i,j #迴圈計數用參數

1. In function “erosion”:

img\_rows, img\_columns #輸入圖檔的行列數

ker\_rows, ker\_columns #kernel的行列數

row\_dist, column\_dist #計算kernel中心距離邊界有多遠，主要目的是看原始圖檔要擴大多少

temp\_img #原始圖檔擴大後的暫存圖檔

new\_img #新圖檔準備接受Erosion後的圖

i,j #迴圈計數用參數

1. In function “binary\_image\_complement”:

img\_rows, img\_columns #輸入圖檔的行列數

new\_img #新圖檔準備接受Erosion後的圖

i,j #迴圈計數用參數

1. In function “hit\_and\_miss\_ur\_corner”:

img\_rows, img\_columns #輸入圖檔的行列數

new\_img #新圖檔準備接受Erosion後的圖

temp\_img1 #原圖檔與kernel\_j進行erosion後的圖

temp\_img2 #原圖的互補圖與kernel\_k進行erosion後的圖

i,j #迴圈計數用參數

1. Outside of function

original\_img #讀取原始圖檔

kernel, kernel\_j, kernel\_k #作業指定會用到的kernel矩陣

Principal code fragment:

#dilation function

def dilation(img, ker):

#獲得輸入圖檔之行列數

img\_rows, img\_columns = img.shape

#獲得kernel之行列數

ker\_rows, ker\_columns = ker.shape

#計算kernel中心距離邊界有多遠，為的是擴大原始圖檔，方便後續迴圈處理

row\_dist, column\_dist = int((ker\_rows-1)/2), int((ker\_columns-1)/2)

#根據上述計算，製作一個比原始圖檔大的暫存圖檔，以img為512\*512, kernel為5\*5來說，暫存圖檔為516\*516，暫存圖檔為往上、往下、往左、往右分別外擴兩列/行，外擴新增的pixel值為0，中間則就是原本輸入圖檔的值

temp\_img = np.zeros((img\_rows+2\*row\_dist, img\_columns+2\*column\_dist), np.int)

temp\_img[row\_dist:img\_rows+row\_dist, column\_dist:img\_columns+column\_dist] = img

#製作一個新圖檔準備接受dilation後的圖

new\_img = np.zeros((img\_rows, img\_columns), np.int)

#進行dilation邏輯計算

for i in range(row\_dist, img\_rows+row\_dist):

for j in range(column\_dist, img\_columns+column\_dist):

if np.any(np.logical\_and(ker, temp\_img[i-row\_dist: i+row\_dist+1, j-column\_dist: j+column\_dist+1])):

new\_img[i-row\_dist, j-column\_dist] = 255

return new\_img

def erosion(img, ker):

#獲得輸入圖檔之行列數

img\_rows, img\_columns = img.shape

#獲得kernel之行列數

ker\_rows, ker\_columns = ker.shape

#計算kernel中心距離邊界有多遠，為的是擴大原始圖檔，方便後續迴圈處理

row\_dist, column\_dist = int((ker\_rows-1)/2), int((ker\_columns-1)/2)

###特別注意這邊外擴pixel的值為1，而在dilation function內是0###

temp\_img = np.ones((img\_rows+2\*row\_dist, img\_columns+2\*column\_dist), np.int)

temp\_img[row\_dist:img\_rows+row\_dist, column\_dist:img\_columns+column\_dist] = img

#製作一個新圖檔準備接受dilation後的圖

new\_img = np.zeros((img\_rows, img\_columns), np.int)

#進行erosion邏輯計算

for i in range(row\_dist, img\_rows+row\_dist):

for j in range(column\_dist, img\_columns+column\_dist):

if not np.any(ker - np.logical\_and(ker, temp\_img[i-row\_dist: i+row\_dist+1, j-column\_dist: j+column\_dist+1])):

new\_img[i-row\_dist, j-column\_dist] = 255

return new\_img

def hit\_and\_miss\_ur\_corner(img, ker\_j, ker\_k):

img\_rows, img\_columns = img.shape

new\_img = np.zeros((img\_rows, img\_columns), np.int)

temp\_img1 = erosion(img, ker\_j)

temp\_img2 = erosion(binary\_image\_complement(img), ker\_k)

for i in range(img\_rows):

for j in range(img\_columns):

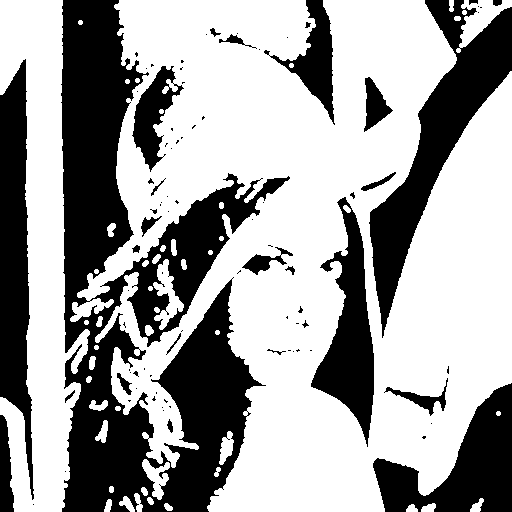
if temp\_img1[i,j]==255 and temp\_img2[i,j]==255:

new\_img[i,j] = 255

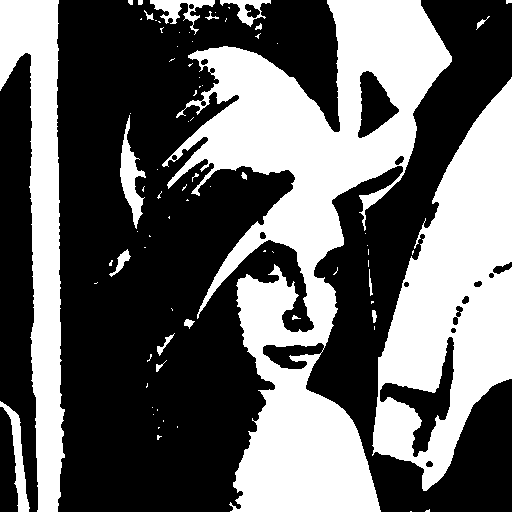
return new\_img

Resulting images

Dilation



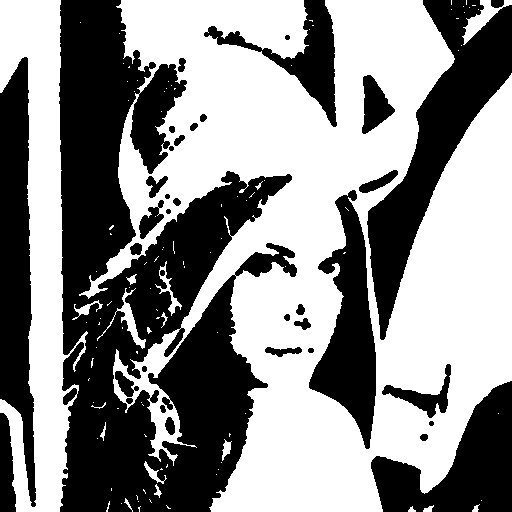
Erosion



Opening



Closing



Hit-and-Miss

