Edge Detection & Morphology

S1154005李育林

Edge Detection

使用sobel kernel對原圖進行kernel運算，分別計算出gx, gy，最後將兩維度之梯度各取平方相加開根號得到梯度值magnitude，即為本圖。

```python
def cal_gradient(image, operator_x, operator_y):
    height, width = image.shape
    ksize = 3
    #image = image.astype(np.float32)
    pad_size = ksize//2
    padded_image = np.pad(image, pad_size, mode = 'reflect')
    gx = np.zeros(image.shape, dtype=np.float32)
    gy = np.zeros(image.shape, dtype=np.float32)

    for y in range(height):
        for x in range(width):
            gx[y, x] = np.sum(padded_image[y:y+ksize, x:x+ksize]*operator_x)
            gy[y, x] = np.sum(padded_image[y:y+ksize, x:x+ksize]*operator_y)
    return gx, gy
```

Gradient計算

Sobel

Edge Detection

與sobel相同作法，但不同之處在於使用的是prewitt kernel

```python
def cal_gradient(image, operator_x, operator_y):
    height, width = image.shape
    ksize = 3
    #image = image.astype(np.float32)
    pad_size = ksize//2
    padded_image = np.pad(image, pad_size, mode = 'reflect')
    gx = np.zeros(image.shape, dtype=np.float32)
    gy = np.zeros(image.shape, dtype=np.float32)

    for y in range(height):
        for x in range(width):
            gx[y, x] = np.sum(padded_image[y:y+ksize, x:x+ksize]*operator_x)
            gy[y, x] = np.sum(padded_image[y:y+ksize, x:x+ksize]*operator_y)
    return gx, gy
```

Gradient計算

Prewitt

Edge Detection

1. 先使用高斯filter降低雜訊圖形後計算圖形之梯度值和角度
2. 接著依照梯度方向和鄰居比較，若小於其中一個鄰居則梯度值設為0(non-maxima suppression)
3. 接著透過高/低threshold去判斷強弱邊
4. 最後檢查弱邊周圍有強邊能連接，有的話則加入邊緣

Canny

Edge Detection

```
for y in range(1, H-1):
  for x in range(1, W-1):
    pixel = G_theta[y,x]
    if((pixel < 67.5 and pixel >= 22.5) or (pixel >= -157.5 and pixel < -112.5)):
      G_dir[y, x] = DIRECTION.SLASH
    elif(pixel < 112.5 and pixel >= 67.5 or (pixel < -67.5 and pixel >= -112.5)):
      G_dir[y, x] = DIRECTION.VERTICAL
    elif(pixel < -22.5 and pixel >= -67.5 or (pixel < 157.5 and pixel >= 112.5)):
      G_dir[y, x] = DIRECTION.BACK_SLASH
    else:
      G_dir[y, x] = DIRECTION.HORIZON
```

判斷梯度方向

```
for y in range(1, H-1):
  for x in range(1, W-1):
    if result_image[y, x] == TL:
      flag = False
      for i in range(-1, 2):
        for j in range(-1, 2):
          if result_image[y+i, x+j] == 255:
            flag = True
            result_image[y][x] = 255
      if not flag:
        result_image[y][x] = 0
```

強弱邊連接

```
#做non-maxima supression
for y in range(1, H-1):
  for x in range(1, W-1):
    cur_dir = G_dir[y, x]
    cur_magnitude = G_magnitude[y, x]
    v1 = 0
    v2 = 0 #鄰居
    if(cur_dir == DIRECTION.VERTICAL):
      v1 = G_magnitude[y+1, x]
      v2 = G_magnitude[y-1, x]
    elif(cur_dir == DIRECTION.HORIZON):
      v1 = G_magnitude[y, x+1]
      v2 = G_magnitude[y, x-1]
    elif(cur_dir == DIRECTION.SLASH):
      v1 = G_magnitude[y+1, x+1]
      v2 = G_magnitude[y-1, x-1]
    elif(cur_dir == DIRECTION.BACK_SLASH):
      v1 = G_magnitude[y-1, x+1]
      v2 = G_magnitude[y+1, x-1]
    else:
      v1 = 255
      v2 = 255

    if(cur_magnitude < v1 or cur_magnitude < v2):
      G_supressed[y, x] = 0
    else:
      G_supressed[y, x] = cur_magnitude
```

Non-maxima supression

Edge Detection

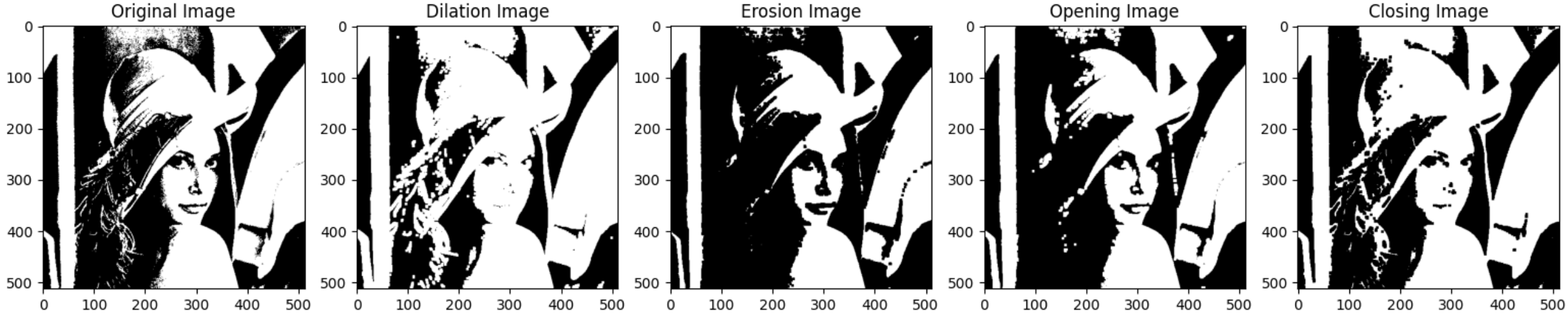
Original

Prewitt

Sobel

Canny

Morphology



*structure element為全為1(255)的3*3圖形

Morphology

將structure element去掃一遍圖，只要和原圖中白色(255)的部分有交集(有碰到)，皆會被轉換成1

```
for y in range(H):
  for x in range(W):
    for i in range(kernel.shape[0]):
      cur_y = y-i+pad_size
      for j in range(len(kernel[i])):
        cur_x = x-j+pad_size
        if cur_y < 0 or cur_y >= H or cur_x < 0 or cur_x >= W:
          continue
        if kernel[i][j] == image[y][x]:
          new_image[cur_y][cur_x] = kernel[i][j]
```

Dilation

Morphology

Erosion實作起來則較dilation複雜，需要整個structure element都在對上原圖都符合才可保留點

```
#一開始設定result_image全部為黑的
#後續判斷時若"kernel有整個在圖內則kernel該點會是255(白的)，而其他部分被erode掉(黑的)"
#new_image = np.full(image.shape, 0)
new_image = np.full((image.shape), 0, dtype=np.uint8)
pad_size = max(len(arr) for arr in kernel)
pad_size = max(pad_size, kernel.shape[0])
pad_size = pad_size//2

pixels = []
for y in range(H):
  for x in range(W):
    flag = True       You, 5 days ago • finish HW1, HW2-canny
    for i in range(kernel.shape[0]):
      cur_y = y+i-pad_size
      for j in range(kernel[i].size):
        cur_x = x+j-pad_size
        if cur_y < 0 or cur_y >= H or cur_x < 0 or cur_x >= W:
          continue
        if image[cur_y][cur_x] != kernel[i][j]:
          flag = False
          break
    if flag == True:
      pixels.append((y,x))
for y, x in pixels:
  new_image[y][x] = 255
```

Erosion

Morphology

即先做erosion後再做dilation，可以用於將不小心因雜訊連接起來的物件分開

Opening

Morphology

即先做dilation後再做erosion，可以用於填補破洞密集的一直在0和255切換的區塊，如這張原圖帽子上方的部分

Opening