[Computer Vision I] Homework 4

學號: R07943087 姓名: 林啟源

Write a program to generate images and histograms:

```python
def main(config):
    # Image Pre-processing
    content = load_image(config.init_pict)  # read in image
    width, height = content.size            # image's width, height
    print("Image width=", width, ", Image height=", height)
    content_np = np.asarray(content).copy() # backup the image
    ret, thresh = cv2.threshold(content_np, 127, 255, cv2.THRESH_BINARY) # do binary morphology

    # octogonal 3-5-5-5-3 kernel
    oct_Kernel = np.ones((5, 5), np.uint8)
    oct_Kernel[0, 0], oct_Kernel[0, 4], oct_Kernel[4, 0], oct_Kernel[4, 4] = 0, 0, 0, 0
    print('')

    # Call Functions
    # (a) Dilation
    Dilation(thresh, oct_Kernel, config)
    # (b) Erosion
    Erosion(thresh, oct_Kernel, config)
    # (c) Opening
    Opening(thresh, oct_Kernel, config)
    # (d) Closing
    Closing(thresh, oct_Kernel, config)
    # (e) Hit-and-miss transform
    HitandMiss(thresh, config)
```

```python
# Image Pre-processing
def load_image(image_path):
    if not os.path.exists(image_path):
        print('Image not exit')
    else:
        image = Image.open(image_path)
        print('Input image:', image_path)
        return image
```

(a) Dilation

```python
# Dilation mask
def Dilation_mask(thresh):
    width, height = thresh.shape
    thresh_ext = np.zeros((width+4, height+4), np.uint8)
    thresh_done = np.zeros((width, height), np.uint8)

    for w in range(2, width+2, 1):
        for h in range(2, height+2, 1):
            if (thresh[w-2][h-2]==255):     # octogonal 3-5-5-5-3 kernel
                thresh_ext[w-2][h-1]=255; thresh_ext[w-2][h+0]=255; thresh_ext[w-2][h+1]=255;
                thresh_ext[w-1][h-2]=255; thresh_ext[w-1][h-1]=255; thresh_ext[w-1][h+0]=255; thresh_ext[w-1][h+1]
                thresh_ext[w+0][h-2]=255; thresh_ext[w+0][h-1]=255; thresh_ext[w+0][h+0]=255; thresh_ext[w+0][h+1]
                thresh_ext[w+1][h-2]=255; thresh_ext[w+1][h-1]=255; thresh_ext[w+1][h+0]=255; thresh_ext[w+1][h+1]
                thresh_ext[w+2][h-1]=255; thresh_ext[w+2][h-0]=255; thresh_ext[w+2][h+1]=255;

    for w in range(width):
        for h in range(height):
            thresh_done[w][h] = thresh_ext[w+2][h+2]

    return(thresh_done)
```

```python
# (a) Dilation
def Dilation(thresh, oct_Kernel, config):
    lena_dilation = Dilation_mask(thresh)
    #lena_dilation = cv2.dilate(thresh, oct_Kernel)
    Image.fromarray(np.uint8(lena_dilation)).save(config.dilation)
    print("Dilation Done")
```

- def Dilation(thresh, oct_Kernel, config) call function “Dilation_mask()”
- “Dilation_mask()” 使用 2 個 for loop 掃描比對 filter 和 binary 圖片，並將 if 判斷後的結果儲存於” thresh_done” np array
- if (thresh[w-2][h-2]==255)，只要目標 pixel=255，便會將周圍的 pixel 數值依照 filter 圖形更改為 255
- thresh_ext = np.zeros((width+4, height+4), np.uint8)會是 516 * 516 大小 np array，經過處理會刪除多餘的邊緣，並儲存於” thresh_done” np array (shape=512*512)

(b) Erosion

```python
# Erosion mask
def Erosion_mask(thresh):
    width, height = thresh.shape
    thresh_ext = np.zeros((width+4, height+4), np.uint8)
    thresh_done = np.zeros((width, height), np.uint8)

    for w in range(width):
        for h in range(height):
            thresh_ext[w+2][h+2] = thresh[w][h]

    for w in range(2, width+2, 1):
        for h in range(2, height+2, 1):     # octogonal 3-5-5-5-3 kernel
            if (thresh_ext[w-2][h-1]==255 and thresh_ext[w-2][h+0]==255 and thresh_ext[w-2][h+1]==255 and \
                thresh_ext[w-1][h-2]==255 and thresh_ext[w-1][h-1]==255 and thresh_ext[w-1][h+0]==255 and thresh_e
                thresh_ext[w+0][h-2]==255 and thresh_ext[w+0][h-1]==255 and thresh_ext[w+0][h+0]==255 and thresh_e
                thresh_ext[w+1][h-2]==255 and thresh_ext[w+1][h-1]==255 and thresh_ext[w+1][h+0]==255 and thresh_e
                thresh_ext[w+2][h-1]==255 and thresh_ext[w+2][h-0]==255 and thresh_ext[w+2][h+1]==255 ):
                thresh_done[w-2][h-2] = 255

    return(thresh_done)
```

```python
# (b) Erosion
def Erosion(thresh, oct_Kernel, config):
    lena_erosion = Erosion_mask(thresh)
    #lena_erosion = cv2.erode(thresh, oct_Kernel)
    Image.fromarray(np.uint8(lena_erosion)).save(config.erosion)
    print("Erosion Done")
```

- def Erosion (thresh, oct_Kernel, config) call function “Erosion_mask ()”
- “Erosion_mask ()” 使用 2 個 for loop 掃描比對 filter 和 binary 圖片，並將 if 判斷後的結果儲存於” thresh_done” np array

- If 條件必須要 filter 覆蓋範圍的 pixel 值都為 255，才會將中心 pixel 值保留為 255
- thresh_ext = np.zeros((width+4, height+4), np.uint8)會是 516 * 516 大小 np array，經過處理會刪除多餘的邊緣，並儲存於" thresh_done" np array (shape=512*512)

(c) Opening

```
# (c) Opening
def Opening(thresh, oct_Kernel, config):
    # erosion followed by dilation
    lena_opening = Erosion_mask(thresh)
    lena_opening = Dilation_mask(lena_opening)
    #lena_opening = cv2.morphologyEx(thresh, cv2.MORPH_OPEN, oct_Kernel)
    Image.fromarray(np.uint8(lena_opening)).save(config.opening)
    print("Opening Done")
```

- erosion followed by dilation:
  lena_opening = Erosion_mask(thresh)
  lena_opening = Dilation_mask(lena_opening)
- #lena_opening = cv2.morphologyEx(thresh, cv2.MORPH_OPEN, oct_Kernel)
  使用 cv2 函數直接產生 Opening 結果
- Image.fromarray(np.uint8(lena_opening)).save(config.opening)
  儲存圖片

(d) Closing

```
# (d) Closing
def Closing(thresh, oct_Kernel, config):
    # Dilation followed by Erosion
    lena_closing = Dilation_mask(thresh)
    lena_closing = Erosion_mask(lena_closing)
    #lena_closing = cv2.morphologyEx(thresh, cv2.MORPH_CLOSE, oct_Kernel)
    Image.fromarray(np.uint8(lena_closing)).save(config.closing)
    print("Closing Done")
```

- Dilation followed by Erosion:
  lena_closing = Dilation_mask(thresh)
  lena_closing = Erosion_mask(lena_closing)
- #lena_closing = cv2.morphologyEx(thresh, cv2.MORPH_CLOSE, oct_Kernel)
  使用 cv2 函數直接產生 Closing 結果
- Image.fromarray(np.uint8(lena_closing)).save(config.closing)
  儲存圖片

(e) Hit-and-miss transform

```python
# (e) Hit-and-miss transform
def HitandMiss(thresh, config):
    width, height = thresh.shape
    # Img A & inv_A
    A_J     = np.zeros((width, height), np.uint8)
    invA_K  = np.zeros((width, height), np.uint8)
    HandM   = np.zeros((width, height), np.uint8)

    # A_J
    for i in range(0, width-1):
        for j in range(1, height):
            if (thresh[i][j-1]==255 and thresh[i][j]==255 and thresh[i+1][j]==255):
                A_J[i][j]=1

    # invA_K
    for i in range(1, width):
        for j in range(0, height-1):
            if (thresh[i-1][j]==0 and thresh[i-1][j+1]==0 and thresh[i][j+1]==0):
                invA_K[i][j]=1

    # HandM
    for i in range(width):
        for j in range(height):
            if (A_J[i][j]==1 and invA_K[i][j]==1):
                HandM[i][j] = 255

    Image.fromarray(np.uint8(HandM)).save(config.ham)
    print("HitandMiss Done")
```

- Use 2 for loop 算出 A 和 J 的 intersection of erosions，A_J[i][j]
  Use 2 for loop 算出 inv A 和 K 的 intersection of erosions，invA_K[i][j]
  Use 2 for loop 算出 A_J[i][j]和 invA_K[i][j]的 intersection of erosions，HandM[i][j]
- Image.fromarray(np.uint8(HandM)).save(config.ham)
  儲存圖片