# [Computer Vision I] Homework 5

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

    # 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(content_np, oct_Kernel, config)
    # (b) Erosion
    Erosion(content_np, oct_Kernel, config)
    # (c) Opening
    Opening(content_np, oct_Kernel, config)
    # (d) Closing
    Closing(content_np, oct_Kernel, 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_init = np.zeros((width+4, height+4), np.uint8)
    thresh_ext = np.zeros((width+4, height+4), np.uint8)
    thresh_done = np.zeros((width, height), np.uint8)

    thresh_ext_init[2:width+2, 2:height+2] = thresh[0:width, 0:height]

    for w in range(2, width+2, 1):
        for h in range(2, height+2, 1):
            max_pixel_v = np.amax(thresh_ext_init[w-2:w+3, h-1:h+2])
            max_pixel_h = np.amax(thresh_ext_init[w-1:w+2, h-2:h+3])
            max_pixel = max(0, max_pixel_v, max_pixel_h)

            # octogonal 3-5-5-5-3 kernel
            thresh_ext[w, h] = max_pixel

    thresh_done[0:width, 0:height] = thresh_ext[2:width+2, 2:height+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()”
- thresh_ext_init[2:width+2, 2:height+2] = thresh[0:width, 0:height]建立一個比原始圖片各邊長多 4 的 np array，並將 gray level 圖片數據存於其中
- 使用 2 個 for loop 掃描比對 filter 和 gray level 圖片，並找出 filter 範圍內 pixel 最大值” max_pixel”
- 將 filter 涵蓋部分中間值的 pixel 值改為” max_pixel”
- thresh_done[0:width, 0:height] = thresh_ext[2:width+2, 2:height+2]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_init = np.ones((width+4, height+4), np.uint8)
    thresh_ext_init = thresh_ext_init * 255
    thresh_ext = np.zeros((width+4, height+4), np.uint8)
    thresh_done = np.zeros((width, height), np.uint8)

    thresh_ext_init[2:width+2, 2:height+2] = thresh[0:width, 0:height]

    for w in range(2, width+2, 1):
        for h in range(2, height+2, 1):
            min_pixel_v = np.amin(thresh_ext_init[w-2:w+3, h-1:h+2])
            min_pixel_h = np.amin(thresh_ext_init[w-1:w+2, h-2:h+3])
            min_pixel = min(255, min_pixel_v, min_pixel_h)

            # octogonal 3-5-5-5-3 kernel
            thresh_ext[w, h] = min_pixel

    thresh_done[0:width, 0:height] = thresh_ext[2:width+2, 2:height+2]
    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 ()”
- thresh_ext_init[2:width+2, 2:height+2] = thresh[0:width, 0:height]建立一個比原始圖片各邊長多 4 的 np array，並將 gray level 圖片數據存於其中
- 使用 2 個 for loop 掃描比對 filter 和 gray level 圖片，並找出 filter 範圍內 pixel 最小值” min_pixel”
- 將 filter 涵蓋部分中間值的 pixel 值改為” min_pixel”
- thresh_done[0:width, 0:height] = thresh_ext[2:width+2, 2:height+2]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