Computer-vision-Homework 5

Mathematical Morphology – Binary Morphology

Due date：16 Nov 2021

Programming language: python 3.9.7

Import lib:

- Opencv: to read and write the image file
- Numpy: to work with the arrays

Original image: lena.bmp

[512(width),512(height),1channel(cv2.IMREAD_GRAYSCALE)]

Code explanation:

```
lena = cv2.imread('lena.bmp', cv2.IMREAD_GRAYSCALE)
```

- imread to load a file with cv2.IMREAD_GRAYSCALE(or using 0 as the parameter of the function)

Q1：Dilation on a grey scale image

```
def dilation(image, kernel):
    dilimage = np.zeros(image.shape, np.uint8)
    m, n = image.shape
    for i in range(m):
        for j in range(n):
            kernel_max = 0
            for position in kernel:
                p, q = position
                if 0 <= (i + p) <= (m - 1) and 0 <= (j + q) <= (n - 1):
                    kernel_max = max(image[i + p][j + q], kernel_max)
```

```
            dilimage[i][j] = kernel_max
    return dilimage
```

We go through each pixel in the original image, and search the octagonal kernel of each pixel to find the **maximum** intensity of those kernel pixels and apply the intensity to the center pixel, which forms a dilation image.

Q2：Erosion on a grey scale image

```
def erosion(image, kernel):
    lena_erosion = np.zeros(image.shape, np.uint8)
    rows, columns = lena_erosion.shape
    for i in range(rows):
        for j in range(columns):
            kernel_min = 255
            for position in kernel:
                p, q = position
                if 0 <= (i + p) < 512 and 0 <= (j + q) < 512:
                    kernel_min = min(image[i + p][j + q], kernel_min)
            lena_erosion[i][j] = kernel_min
    return lena_erosion
```

We go through each pixel in the original image, and search the octagonal kernel of each pixel to find the **minimum** intensity of those kernel pixels and apply the intensity to the center pixel, which forms a dilation image.

Q3：Opening on a grey scale image

```
def opening(image, kernel):
    return dilation(erosion(image, kernel), kernel)
```

Execute erosion first and dilate the grayscale image and we will get the opening image.

Q4：Closing on a grey scale image

```
def closing(image, kernel):
    return erosion(dilation(image, kernel), kernel)
```

Execute dilation first and erode the grayscale image and we will get the closing image.