B06902042 劉愷爲

Computer Vision HW5

Part1 Dilation

$$f : F \to E \text{ and } k : K \to E, \text{ then } f \oplus k : F \oplus K \to E$$

$$(f \oplus k)(x) = \max\{f(x - z) + k(z) | z \in K, x - z \in F\}$$

基本上是利用上面的公式實作，f即灰階圖，k則為kernel也就是35553的八邊形且value = 0， 利用迴圈實作，因為kenel中的value=0，所以我們只要找出每個點周圍的的最大值，範圍則是kernel 35553的部分，先用for找出最大值，並將原點的值換成最大值，存在ans裡。

```
def dilation(img):
    ans = np.zeros((img.shape), np.int)
    for x in range(img.shape[0]):
        for y in range(img.shape[1]):
            maxn = 0
            for idx in kernel:
                if img.shape[0] > x - idx[0] >= 0 and img.shape[1] > y-idx[1] >= 0:
                    if img[x - idx[0], y - idx[1]]  > maxn:
                        maxn = img[x - idx[0], y - idx[1]]
            ans[x, y] = maxn
    return ans
```

Part2 Erosion

$$f : F \to E \text{ and } k : K \to E, \text{ then } f \ominus k : F \ominus K \to E$$

$$(f \ominus k)(x) = \min_{z \in K}\{f(x + z) - k(z)\}$$

參考上面的式子，利用迴圈實作，和dilation很像，只是換成紀錄周遭最小值，並將原點的值 換成最小值，存在ans裡。

```python
def erosion(img, k):
    ans = np.zeros((img.shape), np.int)
    for x in range(img.shape[0]):
        for y in range(img.shape[1]):
            mini = 255
            for idx in kernel:
                if img.shape[0] > x + idx[0] >= 0 and img.shape[1] > y + idx[1] >= 0:
                    if img[x + idx[0], y + idx[1]] < mini:
                        mini = img[x + idx[0], y + idx[1]]
            ans[x, y] = mini
    return ans
```

Part3 Opening

$$f \circ k = (f \ominus k) \oplus k$$

這部分的實作就是先erosion，再dilation

```python
def opening(img):
    ans = erosion(img, kernel)
    ans = dilation(ans)
    return ans
```

Part4 Closing

$$f \bullet k = (f \oplus k) \ominus k$$

這部分的實作就是先dilation，再erosion

```python
def closing(img):
    ans = dilation(img)
    ans = erosion(ans, kernel)
    return ans
```

B06902042 劉愷爲

Dilation

Erosion

Opening

Closing