Computer Vision_HW5

R09922093 楊子萱

In this homework, I use python3 and import cv2 to read and write images and use cv2 function to translate image into the binary image. To run my code, use command line and enter python3 hw5.py (http://hw5.py) .

(a) Dilation

For each pixel, do the “dilation_check” for doing the dilation process.

In the dilation_check function, mask the octagon kernel and set the pixel to the max gray scale value in the kernel.

```
def dilation_check(image, kernel, x, y):
        value = 0
        p_x = x - (kernel.shape[0]//2)
        p_y = y - (kernel.shape[1]//2)
        for i in range(kernel.shape[0]):
                for j in range(kernel.shape[1]):
                        if kernel[i][j] == 1:
                                if p_x + i >=0 and p_x + i < image.shape[0]
                                 and p_y + j >=0 and p_y + j < image.shape[1]:
                                        if image[p_x+i][p_y+j] > value:
                                                value = image[p_x+i][p_y+j]
        return value
def dilation(image, kernel):
        result = np.full((image.shape[0],image.shape[1]),0)
        for i in range(image.shape[0]):
                for j in range(image.shape[1]):
                                result[i][j] = dilation_check(image, kernel, i, j)
        return result
```

(b) Erosion

For each pixel, do the “erosion_check” for doing the erosion process.

In the erosion_check function, mask the octagon kernel and set the pixel to the min gray scale value in the kernel.

```
def erosion_check(image, kernel, x, y):
        value = 256
        p_x = x - (kernel.shape[0]//2)
        p_y = y - (kernel.shape[1]//2)
        for i in range(kernel.shape[0]):
                for j in range(kernel.shape[1]):
                        if kernel[i][j] == 1:
                                if p_x + i >=0 and p_x + i < image.shape[0]
                                 and p_y + j >= 0 and p_y + j < image.shape[1] :
                                        if image[p_x+i][p_y+j] < value:
                                                value = image[p_x+i][p_y+j]
        return value
def erosion(image, kernel):
        result = np.full((image.shape[0],image.shape[1]),0)
        for i in range(image.shape[0]):
                for j in range(image.shape[1]):
                        result[i][j] = erosion_check(image, kernel, i, j)
        return result
```

(c) Opening	**(d) Closing**
erosion then dilation.	dilation then erosion.