Homework 10

Zero Crossing Edge Detection

R09922063 鄭筠庭 資工所碩一

Code explanation:

Extend all the border by 1 pixel. Calculate input pixel gradient magnitude. If the magnitude is larger than threshold, Laplacian output pixel is set to 1. If the magnitude is smaller than negative number of the threshold, Laplacian output pixel is set to -1. In other conditions, Laplacian output pixel is set to 0.

I wrote two functions for all the kernel and threshold. One is to calculate the Laplacian output pixel. The other one is to find the zero-crossing edge.

```
def Laplacian(originImg, rows, columns, threshold, mask, extendNumber):
    image = np.zeros((rows, columns), dtype=int)
    imageBordered = cv2.copyMakeBorder(originImg, extendNumber, extendNumber, extendNumber, extendNumber,
        cv2.BORDER_REFLECT) # Extend image borders
    maskRows = len(mask)
    maskColumns = len(mask[0])

    for r in range(rows):
        for c in range(columns):
            tempSum = 0
            for maskR in range(maskRows):
                for maskC in range(maskColumns):
                    tempSum += int(imageBordered[r+maskR-extendNumber, c+maskC-extendNumber]) * mask[maskR]
                        [maskC]
            if tempSum >= threshold:
                image[r,c] = 1
            elif tempSum <= (-threshold):
                image[r,c] = -1
            else:
                image[r,c] = 0

    return checkCrossing(image, rows, columns, extendNumber)
```

```
def checkCrossing(image, rows, columns, extendNumber):
    result = np.full(image.shape, 255, np.uint8)
    imageBordered = cv2.copyMakeBorder(image, extendNumber, extendNumber, extendNumber, extendNumber, cv2.
        BORDER_REFLECT) # Extend image borders

    for r in range(rows):
        for c in range(columns):
            if image[r,c] != 1:
                continue
            for count in range(len(neighbors)):
                if imageBordered[r+neighbors[count][0]+extendNumber,c+neighbors[count][1]+extendNumber] ==
                    -1:
                    result[r,c] = 0
                    break

    cv2.imshow("Laplacian", result)
    cv2.waitKey(0)
    cv2.destroyAllWindows()
    return result
```

A pixel is declared to have a zero crossing if it is less than –t and one of its eight neighbors is greater than t, or if it is greater than t and one of its eight neighbors is less than –t for some fixed threshold t.

Mask:

(a) Laplace Mask 1: threshold 15

(threshold = 15)

	1	
1	-4	1
	1	

(b) Laplace Mask 2: threshold 15

(threshold = 15)

$\frac{1}{3}$

1	1	1
1	-8	1
1	1	1

(c) Minimum variance Laplacian: 20

$\frac{1}{3}$

2	-1	2
-1	-4	-1
2	-1	2

(d) Laplace of Gaussian: 3000

0	0	0	-1	-1	-2	-1	-1	0	0	0
0	0	-2	-4	-8	-9	-8	-4	-2	0	0
0	-2	-7	-15	-22	-23	-22	-15	-7	-2	0
-1	-4	-15	-24	-14	-1	-14	-24	-15	-4	-1
-1	-8	-22	-14	52	103	52	-14	-22	-8	-1
-2	-9	-23	-1	103	178	103	-1	-23	-9	-2
-1	-8	-22	-14	52	103	52	-14	-22	-8	-1
-1	-4	-15	-24	-14	-1	-14	-24	-15	-4	-1
0	-2	-7	-15	-22	-23	-22	-15	-7	-2	0
0	0	-2	-4	-8	-9	-8	-4	-2	0	0
0	0	0	-1	-1	-2	-1	-1	0	0	0

(e) Difference of Gaussian: 1

```
-1  -3  -4  -6  -7  -8  -7  -6  -4  -3 -1
-3  -5  -8 -11 -13 -13 -13 -11  -8  -5 -3
-4  -8 -12 -16 -17 -17 -17 -16 -12  -8 -4
-6 -11 -16 -16   0  15   0 -16 -16 -11 -6
-7 -13 -17   0  85 160  85   0 -17 -13 -7
-8 -13 -17  15 160 283 160  15 -17 -13 -8
-7 -13 -17   0  85 160  85   0 -17 -13 -7
-6 -11 -16 -16   0  15   0 -16 -16 -11 -6
-4  -8 -12 -16 -17 -17 -17 -16 -12  -8 -4
-3  -5  -8 -11 -13 -13 -13 -11  -8  -5 -3
-1  -3  -4  -6  -7  -8  -7  -6  -4  -3 -1
```

Result:

(a) Laplace Mask1: 15

(b) Laplace Mask2: 15

(c) Minimum variance Laplacian: 20

(d) Laplace of Gaussian: 3000

(e) Kirsch's Compass Operator: 135