

Computer and Robot Vision Homework 10

Zero Crossing Edge Detection

資工碩一 張家源 R07922102

使用語言：Python

Zero Crossing Edge Detector

實作convolution先算出結果

```
def convolution(img, mask):
    mask_size = mask.shape[0]
    res = np.zeros(tuple(np.subtract(img.shape, (mask_size-1, mask_size-1)))).astype(float)
    move = []
    steps = [i for i in range(0, mask_size)]
    for i in steps: #get all movements from steps
        for j in steps:
            move.append((i,j))

    #==convolution==
    for i in range(res.shape[0]):
        for j in range(res.shape[1]):
            grad = 0.0
            for m in move:
                r = i+m[1] #index after movement
                c = j+m[0]
                if r < 0 or r >= img.shape[0] or c < 0 or c >= img.shape[1]:
                    continue
                grad += img[r,c] * mask[m[1], m[0]]
            res[i,j] = grad
    return res
```

檢查Threshold

```
def ThresholdCheck(img, threshold):
    res = np.full(img.shape, 255).astype(float)
    move = []
    steps = [i for i in range(-1, 2, 1)]
    for i in steps: #get all movements from steps
        for j in steps:
            move.append((i,j))
    move.remove((0,0))
    for i in range(img.shape[0]):
        for j in range(img.shape[1]):
            zero_crs = False
            for m in move:
                r = i+m[1]
                c = j+m[0]
                if r < 0 or r >= img.shape[0] or c < 0 or c >= img.shape[1]:
                    continue
                elif img[i,j] > threshold and img[r,c] < -threshold :
                    zero_crs = True

            if zero_crs:
                res[i,j] = 0
    return res
```

Laplacian

```
def Laplacian2(img, threshold ) :  
    mask = np.array([1,1,1,1,-8,1,1,1,1]).astype(float).reshape(3,3)  
    mask = mask/3  
    res = convolution(img, mask)  
    res = ThresholdCheck(res, threshold)  
    cv2.imwrite("Laplacian2.jpg", res)  
    return res
```

Result:



Laplacian2

```
def Laplacian2(img, threshold ) :  
    mask = np.array([1,1,1,1,-8,1,1,1,1]).astype(float).reshape(3,3)  
    mask = mask/3  
    res = convolution(img, mask)  
    res = ThresholdCheck(res, threshold)  
    cv2.imwrite("Laplacian2.jpg", res)  
    return res
```

Result:



Minimum-variance Laplacian

```
def minVarLaplacian(img, threshold ) :
    mask = np.array([2,-1,2,-1,-4,-1,2,-1,2]).astype(float).reshape(3,3)
    mask = mask/3
    res = convolution(img, mask)
    res = ThresholdCheck(res, threshold)
    cv2.imwrite("minimum_Varian_Laplacian.jpg", res)
    return res
```

Result:



Laplacian of Gaussian

```
def LoG(img , threshold):
    mask = np.array([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]).reshape(11,11)
    res = convolution(img, mask)
    res = ThresholdCheck(res, threshold)
    cv2.imwrite("Laplacian_of_Gaussian.jpg", res)
    return res
```

Result:



Difference of Gaussian

```
def DoG(img , threshold):
    mask = np.array([[ -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]])

    res = convolution(img, mask)
    res = ThresholdCheck(res, threshold)
    cv2.imwrite("Differenc_of_Gaussian.jpg", res)
    return res
```

Result:

