Computer Vision Homework 4

R08922079 資工所一 洪浩翔

Part 0

```python
def readImg(filename='lena.bmp'):
    #read img
    image = cv2.imread(filename, cv2.IMREAD_GRAYSCALE)
    print('shape:', image.shape)
    return image

def ImgPreProcess(image, paddingSize, kernelMode):
    #padding
    w, h = image.shape
    binary = np.pad(image, ((paddingSize,paddingSize),(paddingSize,paddingSize)), 'constant',
    print('padding shape:', binary.shape)

    #binarize
    index = np.where(binary >= 128)
    binary = np.zeros(binary.shape)
    binary[index] = 255
    cv2.imwrite('binary.jpg', binary)

    #kernel
    if kernelMode == "Oct":
        kernel = np.array([[0,1,1,1,0],[1,1,1,1,1],[1,1,1,1,1],[1,1,1,1,1],[0,1,1,1,0]])
    else:
        kernel = np.array([[1,1],[0,1]])

    return binary, kernel
```

Image reading and pre-processing

Part 1

(a) Dilation

Binary Lena

Dilation

```python
def dilation(image, kernel):
    #dilation on binary image
    DilKernel = kernel*255
    w,h = image.shape
    dilImg = np.zeros(image.shape)
    for i in range(2, w-2):
        for j in range(2, h-2):
            if image[i,j] == 255:
                dilImg[i-2:i+3, j-2:j+3] = dilImg[i-2:i+3, j-2:j+3] + DilKernel
    index = np.where(dilImg > 0)
    dilImg[index] = 255
    dilImg = dilImg[2:-2, 2:-2]

    return dilImg
```

Code for dilation

(b) Erosion

Binary Lena

Erosion

```python
def erosion(image, kernel):
    #erosion on binary image
    w,h = image.shape
    eroImg = np.zeros(image.shape)
    for i in range(2,w-2):
        for j in range(2,h-2):
            if int(np.sum(image[i-2:i+3, j-2:j+3] * kernel)) == int(np.sum(kernel*255)):
                eroImg[i,j] = 255
    eroImg = eroImg[2:-2, 2:-2]

    return eroImg
```

Code for erosion

(c) Opening

Binary Lena

Opening

```python
def opening(image, kernel):
    #ero than dila
    eroImg = erosion(image, kernel)
    eroImg = np.pad(eroImg, ((2,2),(2,2)), 'constant', constant_values=0)
    openImg = dilation(eroImg, kernel)

    return openImg
```

Code for opening

(d) Closing

Binary Lena

Closing

```python
def closing(image, kernel):
    #dila than ero
    dilImg = dilation(image, kernel)
    dilImg = np.pad(dilImg, ((2,2),(2,2)), 'constant', constant_values=0)
    closImg = erosion(dilImg, kernel)
    return closImg
```

Code for closing

(e) Hit-and-miss transform

Binary Lena

Hit and miss

```python
def HitMiss(image, kernel):
    #Hit-and-miss transform
    invImg = np.ones(image.shape) * 255
    invImg = invImg - image
    w,h = image.shape
    hitAndMissImg = np.zeros(image.shape)
    for i in range(1, w-1):
        for j in range(1, h-1):
            if (int(np.sum(image[i:i+2, j-1:j+1]*kernel)) == int(np.sum(kernel*255)) and
                int(np.sum(invImg[i-1:i+1, j:j+2]*kernel)) == int(np.sum(kernel*255))):
                hitAndMissImg[i,j] = 255
    hitAndMissImg = hitAndMissImg[1:-1, 1:-1]

    return hitAndMissImg
```

Code for hit and miss

```python
if __name__ == "__main__":
    image = readImg()
    binary, kernel = ImgPreProcess(image, 2, "Oct")
    dilImg = dilation(binary, kernel)
    print('dilation shape:', dilImg.shape)
    cv2.imwrite("dilation.jpg", dilImg)
    eroImg = erosion(binary, kernel)
    print('erosion shape:', eroImg.shape)
    cv2.imwrite("erosion.jpg", eroImg)
    openImg = opening(binary, kernel)
    cv2.imwrite("opening.jpg", openImg)
    closImg = closing(binary, kernel)
    cv2.imwrite("closing.jpg", closImg)

    binary, kernel = ImgPreProcess(image, 1, "L")
    hitAndMissImg = HitMiss(binary, kernel)
    print("hit and miss shape:", hitAndMissImg.shape)
    cv2.imwrite("HitAndMiss.jpg", hitAndMissImg)
```

Code for main