Computer Vision HW#7

R08942125 廖克允

1. Binarize Lena

```
src=cv2.imread("lena.bmp",cv2.IMREAD_GRAYSCALE)
#hw2實做過的function
rows,cols=src.shape
srcBinary=np.zeros(shape=src.shape,dtype=src.dtype)
for i in range (rows):
    for j in range(cols):
        if src[i,j]>128:
            srcBinary[i,j]=255
        else:
            srcBinary[i,j]=0
cv2.imwrite("lenaBinary.png",srcBinary)
```

2. Down sampling

```
def downsampling (image,sampleFactor):
    rows,cols=image.shape
    rowsD=int(rows/sampleFactor)
    colsD=int(cols/sampleFactor)
    downsampleImg=np.zeros(shape=(rowsD,colsD),dtype=image.dtype)
    for i in range(rowsD):
        for j in range(colsD):
            downsampleImg[i,j]=image[i*sampleFactor,j*sampleFactor]
    return downsampleImg
```

3. Get neighborhood pixels

```
def getNeighborPixels (image,position):
    positionX,positionY=position
    neighborhoodPixels=np.zeros(9)
    for i in range(3):
        for j in range(3):
            dstX=positionX+(i-1)
            dstY=positionY+(j-1)
            if((0<=dstX<image[0].size) and (0<=dstY<image[1].size)):
                neighborhoodPixels[3*j+i]=image[dstX,dstY]
            else:
                neighborhoodPixels[3*j+i]=0
    neighborhoodPixels=[neighborhoodPixels[4],neighborhoodPixels[5],neighborhoodPixels[1]
                        ,neighborhoodPixels[3],neighborhoodPixels[7],neighborhoodPixels[8]
                        ,neighborhoodPixels[2],neighborhoodPixels[0],neighborhoodPixels[6]]
    return neighborhoodPixels
```

4. F function and h function of Yokoi

```
def fYokoi(a1,a2,a3,a4):
    if ([a1,a2,a3,a4].count('r')==4):
        return str(5)
    else:
        return str([a1,a2,a3,a4].count('q'))
```

```
def hYokoi(b,c,d,e):
    if b==c and (d!=b or e!=b):
        return'q'
    elif b==c and (d==b and e==b):
        return'r'
    elif b!=c:
        return's'
```

5. Count Yokoi connectivity number

```
def yokoiConnectivityNumber(image):
    rows,cols=image.shape
    connectNum = np.chararray(shape=image.shape,unicode=True)
    for i in range(rows):
        for j in range(cols):
            if(image[i,j]!=0):
                neighborhoodPixels= getNeighborPixels(image,(i,j))
                a1= hYokoi(neighborhoodPixels[0],neighborhoodPixels[1],neighborhoodPixels[6],neighborhoodPixels[2])
                a2= hYokoi(neighborhoodPixels[0],neighborhoodPixels[2],neighborhoodPixels[7],neighborhoodPixels[3])
                a3= hYokoi(neighborhoodPixels[0],neighborhoodPixels[3],neighborhoodPixels[8],neighborhoodPixels[4])
                a4= hYokoi(neighborhoodPixels[0],neighborhoodPixels[4],neighborhoodPixels[5],neighborhoodPixels[1])
                connectNum[i,j]=fYokoi(a1,a2,a3,a4)
            else:
                connectNum[i,j]=' '
            if connectNum[i,j]=='0':
                connectNum[i,j]=' '
    return connectNum
```

6. Pair relationship operator

```
def pairRelationshipOperator(YokoiConnectNum):
    """
    input:chararray
    output:chararray
    """
    markedImg= np.chararray(shape=YokoiConnectNum.shape,unicode=True)
    np.pad(YokoiConnectNum,((1,1),(1,1)), 'constant')
    rows,cols=YokoiConnectNum.shape
    for i in range(rows):
        for j in range(cols):
            if YokoiConnectNum[i,j]=='0':
                YokoiConnectNum[i,j]=' '
    for i in range(1,rows-1):
        for j in range(1,cols-1):
            #neighborPixel=getNeighborPixels(YokoiConnectNum,(i,j))
            if(YokoiConnectNum[i,j]==''):
                markedImg[i,j]=' '
            else:
                x1=YokoiConnectNum[i+1,j]
                x2=YokoiConnectNum[i,j-1]
                x3=YokoiConnectNum[i-1,j]
                x4=YokoiConnectNum[i,j+1]
                if (([x1,x2,x3,x4].count('1')>=1) and YokoiConnectNum[i,j]=='1'):
                    markedImg[i,j]='p'
                else:
                    markedImg[i,j]='q'
    return markedImg
```

7. Connected shrink operator

```
def connectedShrinkOperator(markedImg,downsampleImg):
    label = yokoi(downsampleImg)
    rows,cols=markedImg.shape
    for i in range(rows):# from up to down
        for j in range(cols):# from left to right
            if markedImg[i][j] == 'p':# check only if label_pair[i][j] == 'p'
                if label[i][j] == 1:
                    img[i][j] = 0 # if we remove a pixel in img, update the yokoi label
                    label = yokoi(img)
    return img
```

8. Result