HW 7: Thinning

Source Code

All questions are written in Python code, please refer to the file “main.py”.
All images will be stored in the folder “res” (automatically create a new folder).
In accordance with the **FAQ** of course website:

- All parts of the question are written from scratch, except for plotting images

Answer

Steps:

1. Binarize image with threshold 128
2. Downsample binarized image to 8x8
3. Find Yokoi connectivity number on downsampled image
4. Mark pair relationship on mark border-interior image
5. Find thinning image using thinning operator
6. Repeat from step 3 until thinned image and downsample image is the same

Code snippets:

```python
# Mark Pair Relationship Functions ##########################################
def markPair(img):
    pair_img = np.zeros(img.shape, dtype=int)
    for i in range(img.shape[0]):
        for j in range(img.shape[1]):
            if img[i][j] != 1:
                pair_img[i][j] = 0
            else:
                top = down = left = right = False
                if j+1 < img.shape[1]:
                    right = True if img[i][j+1] == 1 else False
                if i-1 > 0:
                    top = True if img[i-1][j] == 1 else False
                if j-1 > 0:
                    left = True if img[i][j-1] == 1 else False
                if i+1 < img.shape[0]:
                    down = True if img[i+1][j] == 1 else False
                if (top or down or left or right):
                    pair_img[i][j] = 1
    return pair_img
#############################################################################
```

```python
# Thinning ##########################################################################
def hShrink(b, c, d, e):
    if (b == c) and (b != d or b != e):
        return 1
    else:
        return 0

def thinning(img, marked):
    pad = np.zeros((img.shape[0]+2, img.shape[1]+2), dtype=int)
    pad[1:-1, 1:-1] = img
    thinned_img = np.zeros(img.shape, dtype=int)
    for i in range(1, pad.shape[0]-1):
        for j in range(1, pad.shape[1]-1):
            if pad[i][j] == 0:
                continue
            a1 = hShrink(pad[i][j], pad[i][j+1], pad[i-1][j+1], pad[i-1][j])
            a2 = hShrink(pad[i][j], pad[i-1][j], pad[i-1][j-1], pad[i][j-1])
            a3 = hShrink(pad[i][j], pad[i][j-1], pad[i+1][j-1], pad[i+1][j])
            a4 = hShrink(pad[i][j], pad[i+1][j], pad[i+1][j+1], pad[i][j+1])
            if ((a1+a2+a3+a4) == 1 and marked[i-1][j-1] == 1):
                thinned_img[i-1][j-1] = 0
                pad[i][j] = 0
            else:
                thinned_img[i-1][j-1] = pad[i][j]
    return thinned_img
#####################################################################################
```

```python
iterations = 1
while True:
    yokoi = yokoiNumber(down)
    pair = markPair(yokoi)
    result = thinning(down, pair)
    # Break if same image, continue otherwise
    if not(np.bitwise_xor(result, down).any()):
        break
    np.savetxt("res/pairRelationship_" + str(iterations) + ".txt", pair, delimiter='', fmt='%s')
    np.savetxt("res/yokoiNumber_" + str(iterations) + ".txt", yokoi, delimiter='', fmt='%s')
    cv2.imwrite("res/thinning_" + str(iterations) + ".bmp", result)
    down = result
    print("Iteration", iterations)
    iterations += 1
```

Thinned images per iteration:

Iteration 1 -> Iteration 6