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:

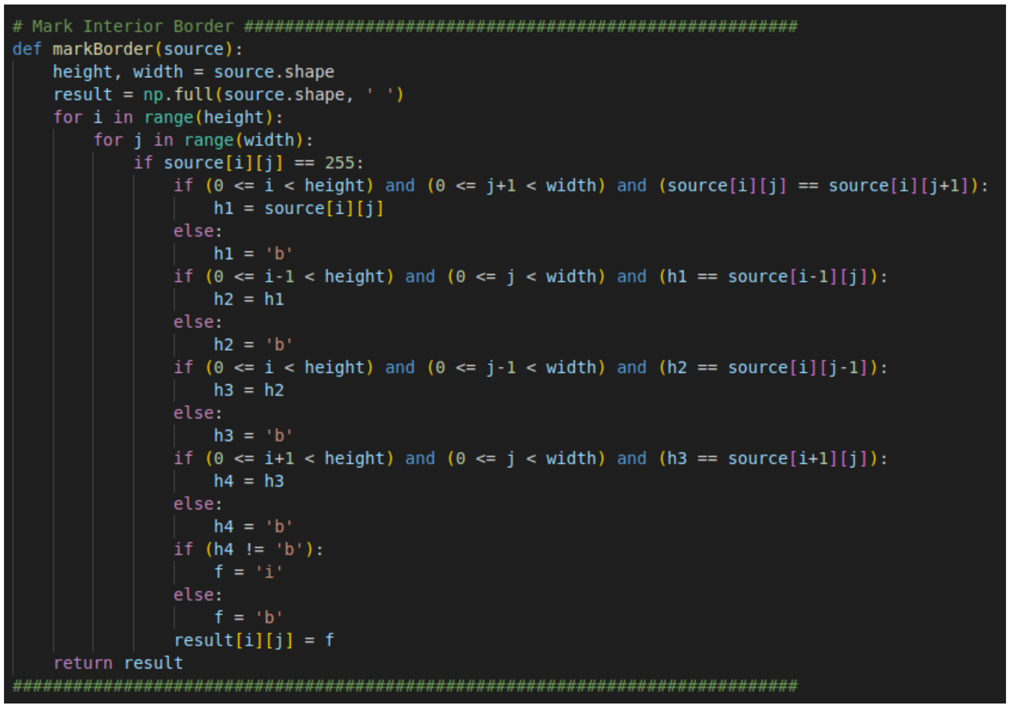
- All parts of the question are written from scratch, except for plotting images

Answer

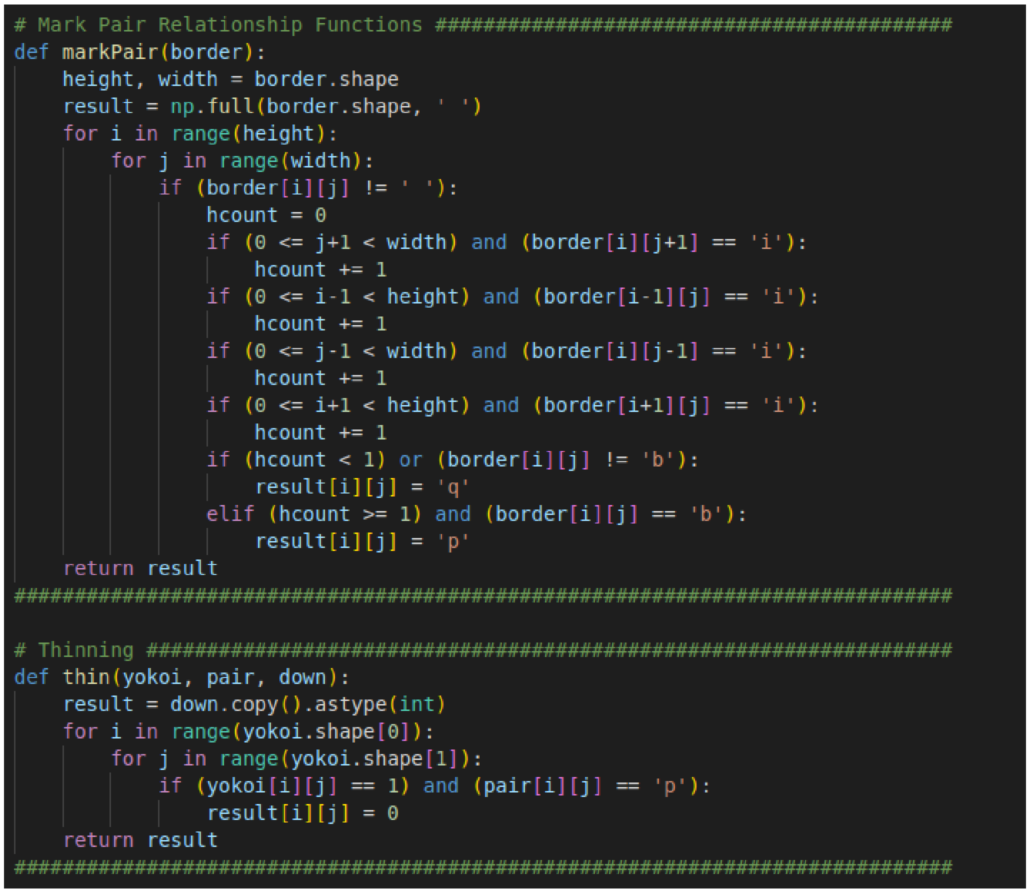
Steps:

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

Code snippets:

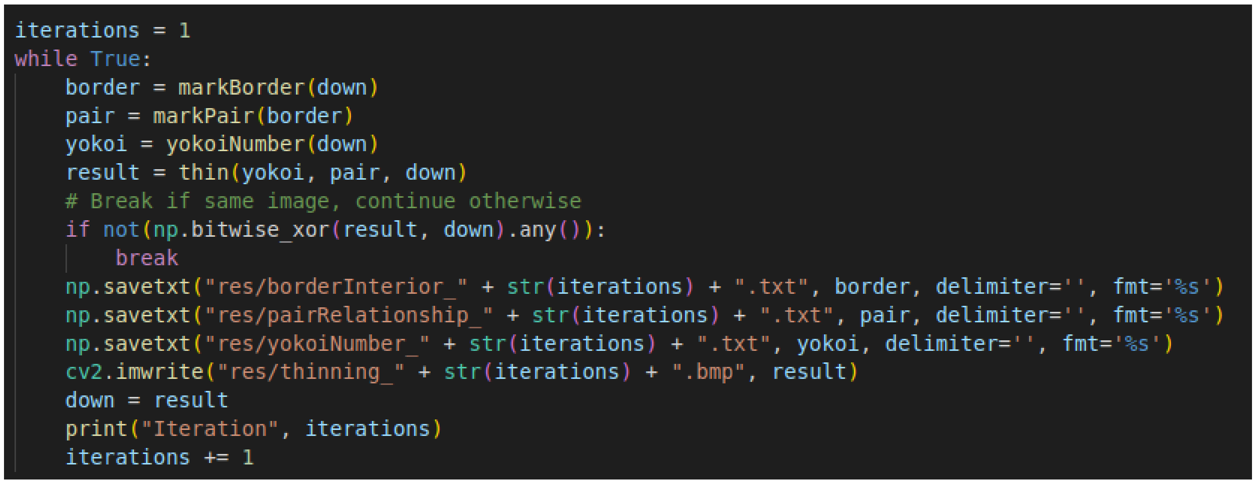


```python
# Mark Interior Border ############################################################
def markBorder(source):
    height, width = source.shape
    result = np.full(source.shape, ' ')
    for i in range(height):
        for j in range(width):
            if source[i][j] == 255:
                if (0 <= i < height) and (0 <= j+1 < width) and (source[i][j] == source[i][j+1]):
                    h1 = source[i][j]
                else:
                    h1 = 'b'
                if (0 <= i-1 < height) and (0 <= j < width) and (h1 == source[i-1][j]):
                    h2 = h1
                else:
                    h2 = 'b'
                if (0 <= i < height) and (0 <= j-1 < width) and (h2 == source[i][j-1]):
                    h3 = h2
                else:
                    h3 = 'b'
                if (0 <= i+1 < height) and (0 <= j < width) and (h3 == source[i+1][j]):
                    h4 = h3
                else:
                    h4 = 'b'
                if (h4 != 'b'):
                    f = 'i'
                else:
                    f = 'b'
                result[i][j] = f
    return result
###################################################################################
```



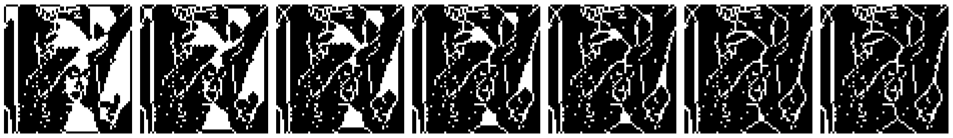
```python
# Mark Pair Relationship Functions ###########################################
def markPair(border):
    height, width = border.shape
    result = np.full(border.shape, ' ')
    for i in range(height):
        for j in range(width):
            if (border[i][j] != ' '):
                hcount = 0
                if (0 <= j+1 < width) and (border[i][j+1] == 'i'):
                    hcount += 1
                if (0 <= i-1 < height) and (border[i-1][j] == 'i'):
                    hcount += 1
                if (0 <= j-1 < width) and (border[i][j-1] == 'i'):
                    hcount += 1
                if (0 <= i+1 < height) and (border[i+1][j] == 'i'):
                    hcount += 1
                if (hcount < 1) or (border[i][j] != 'b'):
                    result[i][j] = 'q'
                elif (hcount >= 1) and (border[i][j] == 'b'):
                    result[i][j] = 'p'
    return result
##############################################################################

# Thinning ###################################################################
def thin(yokoi, pair, down):
    result = down.copy().astype(int)
    for i in range(yokoi.shape[0]):
        for j in range(yokoi.shape[1]):
            if (yokoi[i][j] == 1) and (pair[i][j] == 'p'):
                result[i][j] = 0
    return result
##############################################################################
```



```python
iterations = 1
while True:
    border = markBorder(down)
    pair = markPair(border)
    yokoi = yokoiNumber(down)
    result = thin(yokoi, pair, down)
    # Break if same image, continue otherwise
    if not(np.bitwise_xor(result, down).any()):
        break
    np.savetxt("res/borderInterior_" + str(iterations) + ".txt", border, delimiter='', fmt='%s')
    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 7