Computer Vision 2019 Fall

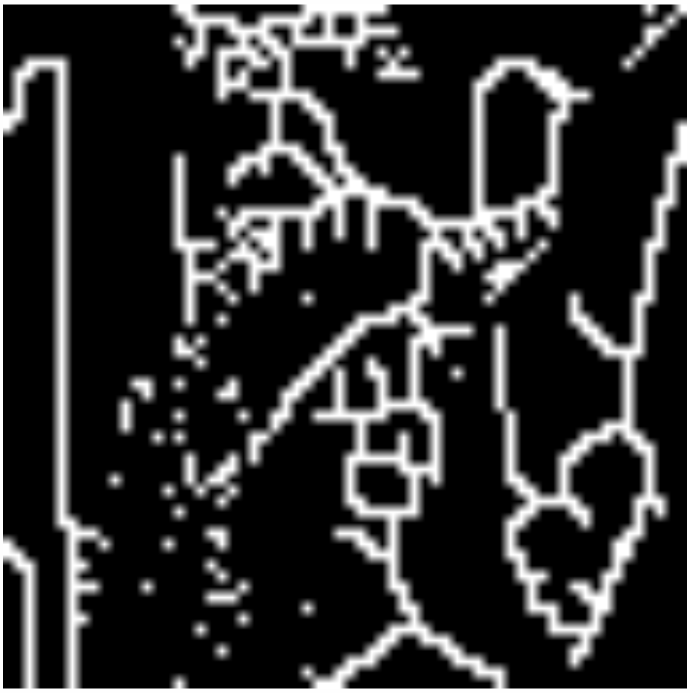
Homework #7

b06902059 資工三 謝宜儒

Description

This homework focuses on thinning on a downsampled image.

Results



Source Code

```
"""
Computer Vision 2019 Fall
Homework #7
"""

import sys
import cv2
import numpy as np

def binarize(img):
  binarized_img = np.zeros((length, width))
  threshold = 128
  for i in range(length):
    for j in range(width):
      if(img[i][j] < threshold):
        binarized_img[i][j] = 0
      else:
        binarized_img[i][j] = 255
  return binarized_img

def downsample(img, scale):
  downsampled_img = np.zeros((int(length / scale), int(width / scale)), dtype
= np.uint8)
  for i in range(0, length, scale):
    for j in range(0, width, scale):
      downsampled_img[int(i / scale)][int(j / scale)] = img[i][j]
  return downsampled_img

def h_yokoi(b, c, d, e):
  if(b == c and (d != b or e != b)):
    return 1
  if(b == c and (d == b and e == b)):
    return 2
  if(b != c):
    return 3

def h_shrink(b, c, d, e):
  if(b == c and (d != b or e != b)):
    return 1
  else:
    return 0

def yokoi_num(img):
  padding_img = np.zeros((img.shape[0] + 2, img.shape[1] + 2), dtype =
np.uint8)
  padding_img[1:-1, 1:-1] = img
  yokoi_img = np.zeros(img.shape, dtype = np.uint8)

  # 1 for q, 2 for r, 3 for s
  for i in range(1, padding_img.shape[0] - 1):
    for j in range(1, padding_img.shape[1] - 1):
      if(padding_img[i][j] == 0):
        continue
      a1 = h_yokoi(padding_img[i][j], padding_img[i][j + 1], padding_img[i -
1][j + 1], padding_img[i - 1][j])
      a2 = h_yokoi(padding_img[i][j], padding_img[i - 1][j], padding_img[i -
1][j - 1], padding_img[i][j - 1])
      a3 = h_yokoi(padding_img[i][j], padding_img[i][j - 1], padding_img[i +
1][j - 1], padding_img[i + 1][j])
      a4 = h_yokoi(padding_img[i][j], padding_img[i + 1][j], padding_img[i +
1][j + 1], padding_img[i][j + 1])
      count_q = (a1 == 1) + (a2 == 1) + (a3 == 1) + (a4 == 1)
      count_r = (a1 == 2) + (a2 == 2) + (a3 == 2) + (a4 == 2)
      if(count_r == 4):
        yokoi_img[i - 1][j - 1] = 5
      else:
        yokoi_img[i - 1][j - 1] = count_q
  return yokoi_img

def pair_relationship(img):
  padding_img = np.zeros((img.shape[0] + 2, img.shape[1] + 2), dtype =
np.uint8)
  padding_img[1:-1, 1:-1] = img
  pair_relationship_img = np.zeros(img.shape, dtype = np.uint8)
  # 0 for q, 1 for p
  for i in range(1, padding_img.shape[0] - 1):
    for j in range(1, padding_img.shape[1] - 1):
      if(padding_img[i][j] != 1):
        pair_relationship_img[i - 1][j - 1] = 0
      else:
        if(padding_img[i][j + 1] == 1 or padding_img[i - 1][j] == 1 or
          padding_img[i][j - 1] == 1 or padding_img[i + 1][j] == 1):
          pair_relationship_img[i - 1][j - 1] = 1
  return pair_relationship_img

def thinning(img, marked_img):
  padding_img = np.zeros((img.shape[0] + 2, img.shape[1] + 2), dtype =
np.uint8)
  padding_img[1:-1, 1:-1] = img
  thinned_img = np.zeros(img.shape, dtype = np.uint8)

  for i in range(1, padding_img.shape[0] - 1):
    for j in range(1, padding_img.shape[1] - 1):
      if(padding_img[i][j] == 0):
        continue
      a1 = h_shrink(padding_img[i][j], padding_img[i][j + 1], padding_img[i -
1][j + 1], padding_img[i - 1][j])
      a2 = h_shrink(padding_img[i][j], padding_img[i - 1][j], padding_img[i -
1][j - 1], padding_img[i][j - 1])
      a3 = h_shrink(padding_img[i][j], padding_img[i][j - 1], padding_img[i +
1][j - 1], padding_img[i + 1][j])
      a4 = h_shrink(padding_img[i][j], padding_img[i + 1][j], padding_img[i +
1][j + 1], padding_img[i][j + 1])
      if((a1 + a2 + a3 + a4) == 1 and marked_img[i - 1][j - 1] == 1):
        thinned_img[i - 1][j - 1] = 0
        padding_img[i][j] = 0
      else:
        thinned_img[i - 1][j - 1] = padding_img[i][j]
  return thinned_img

def plot_img(img, filename, title):
  cv2.imwrite(filename, img)
  cv2.imshow(title, img)
  cv2.waitKey(0)
  cv2.destroyAllWindows()
```

```
if __name__ == "__main__":
  if(len(sys.argv) != 2):
    printf("Usage: python3 hw7.py [input image]\n")

  # Read the input image in grayscale mode
  img = cv2.imread(sys.argv[1], cv2.IMREAD_GRAYSCALE)
  length, width = img.shape[0], img.shape[1]

  binarized_img = binarize(img)
  downsampled_img = downsample(binarized_img, 8)
  # thinning until no change
  while(1):
    yokoi_img = yokoi_num(downsampled_img)
    marked_img = pair_relationship(yokoi_img)
    thinned_img = thinning(downsampled_img, marked_img)
    if((thinned_img == downsampled_img).all()):
      break
    downsampled_img = thinned_img

  plot_img(thinned_img, 'thinned.bmp', 'thinned_img')
```

To run the source code, type the following line in a terminal:

```
python3 hw7.py [input image]
```

where in this homework, the input image is **lena.bmp**.