Computer Vision
Homework 6 Report

林義聖
B03902048

November 23, 2016

Introduction

In this homework assignment, we're going to calculate *Yokoi Connectivity Number*.

I use *Python* as my programming language and *Pillow* as my image library. It is a fork of *PIL*, which is the original image library of *Python*. And I use *Pillow* for reading input image, and transfering image data into *List* of *Python*.

Figure 1: original lena.bmp

Program Structure

There's only one program in my submission, and I use program parameters to decide which algorithm we want to apply on the input image. All the things done in my program is:

1. Import required library
2. Open input image
3. Binarize the image
4. Downsample the image
5. Calculate Yokoi connectivity number
6. Output matrix to text file

```
import sys
from PIL import Image

# Setup system parameters
...

def binarize(data):
  ...

def downsampling(data, hei, wid):
  ...

def get_pattern(a1, a2, a3):
  ...

def yokoi(data, hei, wid):
  ...

def main():
  # get input image
  ...

  # get 1D image data
  data_seq = list(in_img.getdata())

  # binarize the image
  bin_data = binarize(data_seq)

  # downsampling the image
  down_data, newhei, newwid = downsampling(bin_data, height, width)

  # yokoi neighborhood operate
  out_data = yokoi(down_data, newhei, newwid)

  # open output file and save
  ...
```

```
37
38 if __name__ == '__main__':
39     main()
```

Binarize

We need to binarize the input image first. I binarize the image with threshold 127, and for my convenience, I simply binarize the image data to 1 or 0.

```
1 def binarize(data):
2   result = []
3   for p in data:
4     result.append(1 if p > 127 else 0)
5   return result
```

Downsampling

After binarizing the image, I downsample the image by choosing the topmost-left point of a 8×8 matrix.

```
1 def downsampling(data, hei, wid):
2   result = []
3   offset = 8
4   for y in range(0, hei, offset):
5     for x in range(0, wid, offset):
6       result.append(data[y * wid + x])
7 return (result, int(hei/offset), int(wid/offset))
```

Figure 2: downsampled lena.bmp

Yokoi Connectivity Number

To calculate Yokoi connectivity number, I use the following steps:

1. Expand the data matrix, append topmost, bottommost row and left-most, rightmost column to it
2. For pixels of original data matrix, get the pattern of its neighbors
3. Calculate Yokoi connectivity number from patterns

4. Keep doing step 2 and 3 to the end

```
def get_pattern(a1, a2, a3):
    if a1 == 1:
        if a2 == 1 and a3 == 1:
            return 'r'
        else:
            return 'q'
    else:
        return 's'

def yokoi(data, hei, wid):
  expd = [0] * (hei+2) * (wid+2)
  for y in range(hei):
    for x in range(wid):
      expd[(y+1) * (wid+2) + (x+1)] = data[y * wid + x]

  result = [' '] * (hei * wid)
  f = {'r':0, 'q':0, 's':0}
  for y in range(1, hei+1):
    for x in range(1, wid+1):
      f['r'] = 0
      f['q'] = 0
      f['s'] = 0
      if expd[y * (wid+2) + x] == 1:
        f[get_pattern(expd[y*(wid+2)+(x+1)],expd[(y-1)*(wid+2)+(x+1)],expd
            [(y-1)*(wid+2)+x])] += 1
        f[get_pattern(expd[(y-1)*(wid+2)+x],expd[(y-1)*(wid+2)+(x-1)],expd[
            y*(wid+2)+(x-1)])] += 1
        f[get_pattern(expd[y*(wid+2)+(x-1)],expd[(y+1)*(wid+2)+(x-1)],expd
            [(y+1)*(wid+2)+x])] += 1
        f[get_pattern(expd[(y+1)*(wid+2)+x],expd[(y+1)*(wid+2)+(x+1)],expd[
            y*(wid+2)+(x+1)])] += 1
      if f['r'] == 4:
        result[(y-1) * wid + (x-1)] = str(5)
      else:
        result[(y-1) * wid + (x-1)] = str(f['q'])
    else:
      result[(y-1) * wid + (x-1)] = ' '
  return result
```

Result

How to Use

There's only one executable program in my submission, and its name is *yokoi.py*. To run this program, just type this command "`./yokoi.py [input image] [output file]`".