R09922190 王祥任

CV Homework 1

Abstract

本次作業要求對圖片完成以下像素操作，(1) 上下翻轉、(2) 左右翻轉、(3) 對角線翻轉、(4) 順時鐘旋轉45度、(5) 縮小為一半和(6) 以128為分界二值化圖片等。其中，(第一部分)前三項要求除圖片讀取與輸出，不可使用任何套件，而(第二部分)後三項並無特殊規定，以任意軟體或套件完成即可。

Implementation

Environment

- Programming Language: Python3
- Python Package: opencv-python, matplotlib, numpy
- Execution: `python3 hw1-main.py`

在執行時，請務必將 lena.bmp 放置在和 main.py 相同的檔案目錄下，並確認檔名相同。三種套件中，opencv-python用於讀取和寫出圖片，並有使用部分函數於第二部分(可使用任意軟體和套件)，matplotlib僅用於runtime時即時顯示圖片，須配合jupyter使用，在此不贅述，而最後的numpy則僅用於第二部分中計算pixel數值。

Flips

本章節會直接示範第一部分所有的實作，包含以下：

```
 9  def my_flip(lena, tag='ud'):
10      after = copy.deepcopy(lena)
11      for r in range(lena.shape[0]):
12          for c in range(lena.shape[1]):
13              if (tag=='ud'): # upside-down
14                  after[r][c] = lena[lena.shape[0]-1-r][c]
15              elif(tag=='rl'): # right-side-left
16                  after[r][c] = lena[r][lena.shape[1]-1-c]
17              elif(tag=='dg'): # diagonally flip
18                  after[r][c] = lena[c][r]
19              else: # no flip
20                  after[r][c] = lena[r][c]
21
22      return after
```

- 上下翻轉 (upside-down) (line 13、14)
- 左右翻轉 (right-side-left) (line 15、16)
- 對角線翻轉 (diagonally flip) (line 17、18)

三者概念差不多，都是藉由兩層for-loop來遍歷所有pixel，而

1. 上下反轉，需要做的是調換行(row)的順序，由於該lena.bmp的高度為512個pixel，故原先的row(0)則變為row(511)，而原先的row(1)則變為row(510)，依此類推即可。

2. 左右反轉，則是調換列(column)的順序，由於該lena.bmp的寬度為512個pixel，故原先的column(0)則變為column(511)，而原先的column(1)則變為column(510)，依此類推即可。

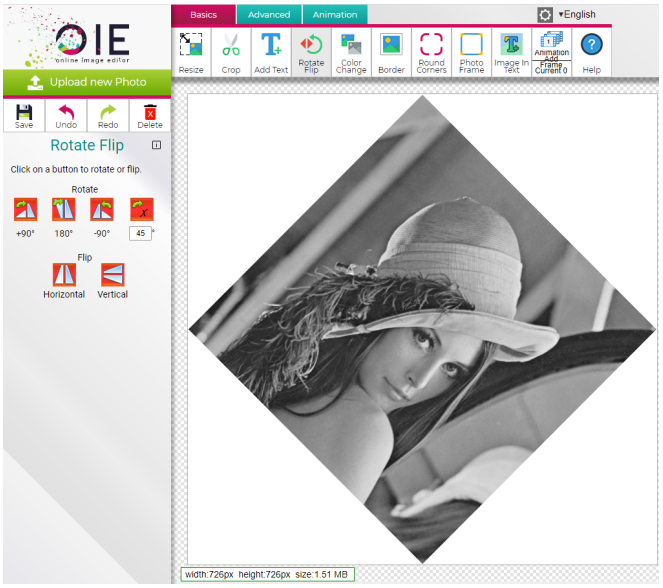
3. 對角線翻轉，則是類似轉置矩陣，且多虧此次獨入的圖片為正方形(512x512)，故將pixel的row index和column index對調即可。

下方圖組即為最終呈現的成果，左上角的圖片為原圖(為了排版呈現，有修改顯示尺寸)：

<table>
<tr>
<td> 
Original</td>
<td> 
Upside-Down</td>
</tr>
<tr>
<td> 
Right-Side-Left</td>
<td> 
Diagonally Flip</td>
</tr>
</table>

Rotate 45 Degrees Clockwise

從此部分開始，允許使用軟體來操作圖片，而我是選用網路上的[圖片編輯網站]，在上傳團片之後，根據下方圖片的操作即可。選定[Rotate Flip]，輸入旋轉角度45度，預設即順時鐘旋轉。



OIE 操作介面

而最終的成果呈現如下，左側圖片為原圖，同樣為了排版呈現，有修改顯示尺寸：

	
Original	Rotate 45 Degrees Clockwise

Shrink In Half

同樣在此部分，允許使用既有軟體和套件來操作圖片。我們要做的是將兩邊長各縮為一半。

```
40  # Shrink
41  def shrink(lena, target_size=(256,256)):
42      after = copy.deepcopy(lena)
43      after = cv2.resize(lena, target_size)
44      return after
45
46  shrink_lena = shrink(lena, (lena.shape[0]//2, lena.shape[1]//2))
47  plt.imshow(shrink_lena)
48  cv2.imwrite('shrink lena.bmp',shrink_lena)
49  cv2.imwrite('shrink lena.png',shrink_lena)
```

那直接利用opencv提供的resize函數，插值的方式直接使用預設即可。而最終成果如下圖組，上圖為原圖：

Original (512x512)

Shrink In Half (256x256)

Binarize at 128

同樣在此部分，允許使用既有軟體和套件來操作圖片。我們要做的是以灰階值128為界，將圖片二值化為0或255。因為opencv讀檔預設為RGB模式，故每個像素是有三個值得數組，但其實該圖片為灰階圖片，所以三的值會是相同的。而為求保險，我採用的計算方式為將數組中的值總合起來，若小於128*3，則該像素轉為0，否則轉為255。

```
51  # Binarize
52  def binarize(lena, thr=128):
53      after = copy.deepcopy(lena)
54      for r in range(after.shape[0]):
55          for c in range(after.shape[1]):
56              if (np.sum(after[r][c]) < thr*3) : after[r][c] = [0,0,0]
57              else: after[r][c] = [255,255,255]
58      return after
59
60  bina = binarize(lena)
61  plt.imshow(bina)
62  cv2.imwrite('binarize lena.bmp',bina)
63  cv2.imwrite('binarize lena.png',bina)
```

而以下為最終呈現成果，左圖為原圖，有因排版因素修改顯示尺寸：

	
Original	Binarize at 128