NTNU 影像處理 HW3

廖家緯

2020.3.31

- **Outline**: For each pixel $I(x, y)$,

  1) Calculate quantization error

$$E(x,y) = I(x,y) = \begin{cases} I(x,y), & \text{if } I(i,j) < 128 \\ I(x,y) - 255, & \text{if } I(i,j) \geq 128 \end{cases}.$$

  2) Spread the error according to Floyd-Steinberg

     i.e.,

	(x,y)	$+\frac{7}{16}E$
$+\frac{3}{16}E$	$+\frac{5}{16}E$	$+\frac{1}{16}E$

  3) Quantize new $I(x, y)$ to 0 or 255 using 128 as the threshold

- **Code(Python):**

```python
# coding: utf-8

#引入模組
import numpy as np
import cv2

# 讀取灰階影像
I = cv2.imread('Gray.jpg', cv2.IMREAD_GRAYSCALE)

#判斷影像的矩陣大小
n = np.shape(I)[0]
m = np.shape(I)[1]
E = np.zeros((n,m), np.double)
I = np.double(I)
```

```
15
16  #For each pixel I(x,y)
17  for i in range(n):
18      for j in range(m):
19
20          #1)Calculate quantization error
21          if I[i, j] >= 128:
22              E[i, j] = I[i, j]-255
23          else:
24              E[i, j] = I[i, j]
25
26          #2)Floyd-Steinberg
27          if j+1<=m-1:
28              I[i, j+1] += (7/16)*E[i, j]
29          if i+1<=n-1 and j-1>=0:
30              I[i+1, j-1] += (3/16)*E[i, j]
31          if i+1<=n-1:
32              I[i+1, j] += (5/16)*E[i, j]
33          if i+1<=n-1 and j+1<=m-1:
34              I[i+1, j+1] += (1/16)*E[i, j]
35
36  #3)Quantize new I(x,y) to 0 or 255 using 128 as the threshold
37  for i in range(n):
38      for j in range(m):
39          if I[i, j]>=128:
40              I[i, j] = 1
41          else:
42              I[i, j] = 0
43  I = I*255
44  I = np.uint8(I)
```

- **Result:**

原圖

Floyd-Steinberg

- **Experience:**

  這次作業花了很多時間 debug。最初我誤解老師題目的意思，我先將 E 矩陣建完後才去更新每個 I 矩陣，也就是分別用兩個雙重迴圈完成 E 和 I，然而做出來的結果非常怪異，因此我反覆重看老師的題目後，才發現兩件事情要在同一個迴圈做。而第二個令我困擾的問題是我一開始忘了將 I 轉型，導致我做出來的碎片感很重，事實上我對各種的型態還不太熟 (沒修過資工的計概)，所以只能上網查資料，一個一個嘗試，花我不少的功夫才完成。另外，這次的作業可以明顯感覺我的方法需要跑大約 3 到 5 秒的時間，雙重迴圈 ($\mathrm{O}(n^2)$) 的時間複雜度讓程式執行的時間上升許多，之後我想嘗試用一些矩陣計算的技巧進行優化，降低程式執行的時間複雜度。