

# Computer Vision HW1 Report

---

B05901011 許秉倫

## 1. Implement

### 1. Convert to grey images

將不同的weight乘上bgr值，接著利用 `cv2.imwrite` 將他們存入 `grey/` 中

### 2. Bilateral filter

filter這一部分重點是要盡量使用numpy的內建function，因為他底層是用c++實作的，跑起來會比自己用python刻快非常非常多(我一開始用python寫一張圖就要跑個一分鐘...)。

幾個重要的trick是，第一，window內部的元素相乘時使用element-wise的相乘，把兩個正方形直接乘在一起、第二，最後取平均的時候使用 `np.average` 這個函式，他會自動幫你把你的Intensity乘上weight然後取平均，效能非常好。

在這裡我自己寫的時候搞錯了兩件事情，第一，filter完的圖片應是和原圖經過filter的結果相減，而不是直接相減，因為這樣可以使noise縮小、第二，sigma\_r是標準化後的標準差，我一開始直接把他丟進去算，結果產生了有好幾個第一名的情況，後來經過助教的協助，才發現原來要把他乘上255倍，之後結果就正常多了。(感謝助教<(\_\_)>)

### 3. Local min selection

Voting這個部分如果知道他其實是在平面上取極值後，就會發現實際上不難，關鍵的點是要知道一個點的旁邊最多有六位鄰居(+0.1 -0.1 0)的排列，然後以key為tuple(表示他的weights)，value為delta的形式存入一個dictionary中，接著，iterate這個函式，對於每個點看他旁邊六個鄰居，不存在的鄰居跳過，如果有其中一個比他小，那他就不是local min。

這部分容易犯下的錯是要對小數點取round，不然在字典中會找不到key，導致有一大堆local min的情況出現。

## 2. Usage

---

### 1. Convert to grey image

執行後會將所有轉換後的灰階圖片存入 `grey/` 中

```
python3 hw1-1.py [image_name]
# ex:
python3 hw1-1.py 2a
```

### 2. Bilateral filter

1. 執行後會將所有filter後的結果存入 `filter/` 中，voting結束後，將此次排行結果存入 `scoreboard/`





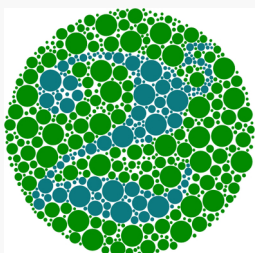
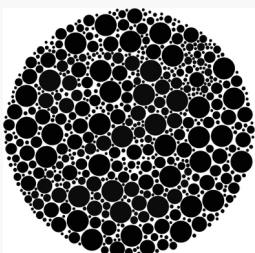
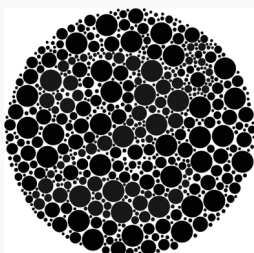
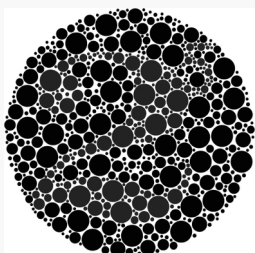




```
python3 hw1-2.py [image_name]
# ex:
python3 hw1-2.py 2a
```

### 3. Run group2 directly

1. 此sh檔會自動幫你跑完group2所有的圖

```
./run2.sh
```

## 3. Result(Group 2)

原圖	Rank1	Rank2	Rank3
2a	(0,1,0): 9票	(1,0,0):7票	(0,0,1):7票
			
2b	(0,0,1): 6票	(0.1,0,0.9): 1票	(0.2,0,0.8):1票
			
2c	(0,0,1): 7票	(0,0.1,0.9): 2票	(0,0.2,0.8): 0票
			

1. 2b組其實有三張第二名，未列上來的為(0.3, 0, 0.7)
2. 2c組第三名從缺，因此我用肉眼挑一張表現最好的當作第三名