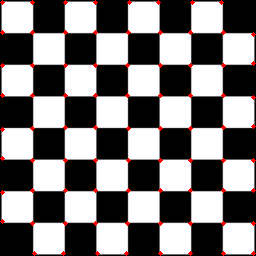
Name: 梁中瀚

Student ID: r09922a02

# Part 1

Detected corner for 1.png



Detected corner for 2.png



Detected corner for 3.png



Threshold 25 on 2.png



Threshold 50 on 2.png



Threshold 100 on 2.png



三種threshold的差異

Threshold越小回傳的corner就越多。

Threshold從25變成50，比較明顯的變化在背景藍色的波浪。

Threshold從50變成100，就沒有太多明顯的減少。

比較特別的是飛機上的字體，不管threshold設多少，字體的corner都會被偵測到，可能是相對較穩定的特徵點。

# Part 2

1.png

Cost for each filtered image;

|  |  |
| --- | --- |
| RGB轉gray scale參數 | L1 norm大小 |
| CV2.cvtColor | 1207799 |
| 0.1,0.0,0.9 | 1439568 |
| 0.2,0.0,0.8 | 1305961 |
| 0.2,0.8,0.0 | 1393620 |
| 0.4,0.0,0.6 | 1279697 |
| 1.0,0.0,0.0 | 1127913 |

Original RGB image:



Lowest cost gray scale image:



Lowest cost filtered RGB image:



Highest cost gray scale image:



Highest cost filtered RGB image:



Difference between two gray scale images:

很明顯，l1 norm數值較小的gray scale圖，葉子與背景雜草的差異較大。

但是我其實看不出來joint bilateral filter作用過的RGB圖片的差別。

2.png

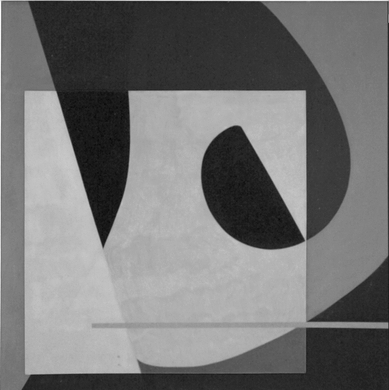
Cost for each filtered image;

|  |  |
| --- | --- |
| RGB轉gray scale參數 | L1 norm大小 |
| CV2.cvtColor | 183851 |
| 0.1,0.0,0.9 | 77884 |
| 0.2,0.0,0.8 | 86023 |
| 0.2,0.8,0.0 | 188019 |
| 0.4,0.0,0.6 | 128341 |
| 1.0,0.0,0.0 | 110862 |

Original RGB image:



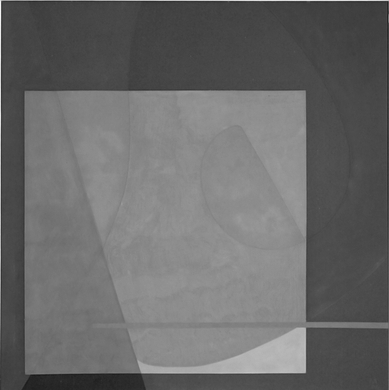
Lowest cost gray scale image:



Lowest cost filtered RGB image:



Highest cost gray scale image:



Highest cost filtered RGB image:



Difference between two gray scale images:

最高cost的灰階圖，最外層的正方形幾乎變成同一個數值了，用肉眼很難看出區別，相反的，cost最小的灰階圖就可以用肉眼區分開來。

Cost最高的灰階圖，橘色跟紫色感覺被map到很相近的灰階，所以造成小正方形內的區塊顏色相近，而cost最低的灰階圖，雖然能夠將橘色紫色分很開，但是藍色和紫色又非常接近。

How to speed up the implementation of bilateral filter:

先將原圖轉換成(kernel移動範圍 \* kernel size)的矩陣，再把kernel也轉換成(1 \* kernel size)的向量，這兩個相乘就會是做完convolution的結果。所以for loop只會出現在建立(kernel移動範圍 \* kernel size)的矩陣的時候。