









Computer Vision HW1 Report

Student ID: R12921a10

Name: 謝宗翰

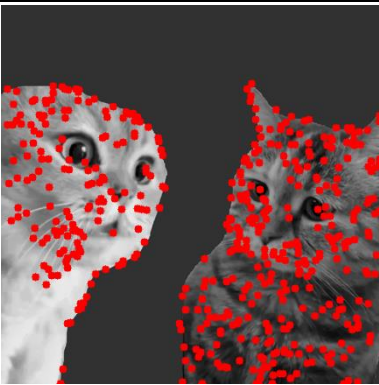


Part 1.

- Visualize the DoG images of 1.png.

| | DoG Image (threshold = 3) | | DoG Image (threshold = 3) |
|------------|---|------------|---|
| DoG1-1.png |  | DoG2-1.png |  |
| DoG1-2.png |  | DoG2-2.png |  |
| DoG1-3.png |  | DoG2-3.png |  |
| DoG1-4.png |  | DoG2-4.png |  |

- Use three thresholds (1,2,3) on 2.png and describe the difference.

| Threshold | Image with detected keypoints on 2.png |
|-----------|--|
|-----------|--|

| | | |
|---|---|--|
| 1 |  | |
| 2 |  | |
| 3 |  | |

由上面幾張圖的結果可知，當 threshold 增加時，key points 會減少。threshold = 1 時，key points 超級多，而且蠻不準的，很多都是在亂標。Threshold = 3 時，邊緣處理得比較精準，key points 不會像 1 時這麼亂。






Part 2.

- **Report the cost for each filtered image.**




| Gray Scale Setting | Cost (1.png) |
|---------------------|--------------|
| cv2.COLOR_BGR2GRAY | 1207799 |
| $R*0.0+G*0.0+B*1.0$ | 1439568 |
| $R*0.0+G*1.0+B*0.0$ | 1305961 |
| $R*0.1+G*0.0+B*0.9$ | 1386209 |
| $R*0.1+G*0.4+B*0.5$ | 1277424 |
| $R*0.8+G*0.2+B*0.0$ | 1127895 |

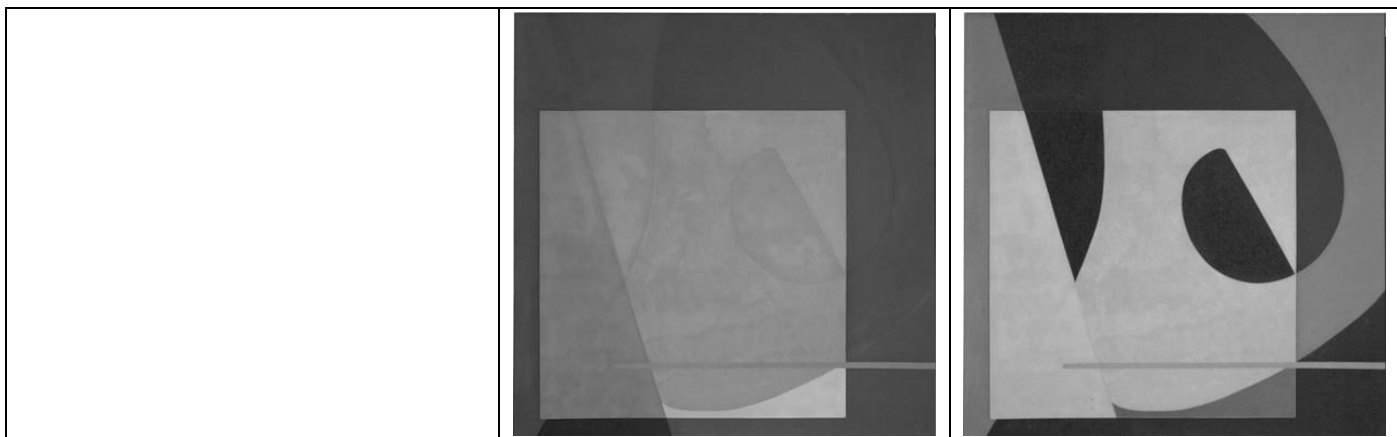
| Gray Scale Setting | Cost (2.png) |
|---------------------|--------------|
| cv2.COLOR_BGR2GRAY | 183851 |
| $R*0.1+G*0.0+B*0.9$ | 78454 |
| $R*0.2+G*0.0+B*0.8$ | 86422 |
| $R*0.2+G*0.8+B*0.0$ | 187520 |
| $R*0.4+G*0.0+B*0.6$ | 128825 |
| $R*1.0+G*0.0+B*0.0$ | 110862 |

- Show original RGB image / two filtered RGB images and two grayscale images with highest and lowest cost.

| Original RGB image (1.png) | Filtered <u>RGB image</u> and <u>Grayscale image</u> of Highest cost | Filtered <u>RGB image</u> and <u>Grayscale image</u> of Lowest cost |
|---|---|--|
|  |  |  |
| |  |  |

從 cost 最高的 gray image 來看，很不好看出楓葉與旁邊的草。而從 cost 最低的 gray image 來看，可以輕易地分辨楓葉和旁邊的草。這表示由權重 (0.8, 0.2, 0.0) 生成的 gray image 是將 rgb image 轉換成灰階影像的最好的參數。

| Original RGB image (2.png) | Filtered <u>RGB image</u> and <u>Grayscale image</u> of Highest cost | Filtered <u>RGB image</u> and <u>Grayscale image</u> of Lowest cost |
|---|--|---|
|  |  |  |



從 cost 最高的 gray image 來看，這個酷酷的抽象畫的紋理變得很淡，不好看出來。相較之下，cost 最低的 gray image 來看，紋理非常明顯。這表示由權重 (0.1, 0.0, 0.9) 產生的 gray image 是將 rgb image 轉換成灰階影像的最好的參數。

- **Describe how to speed up the implementation of bilateral filter.**

原本的方法：

用 for 迴圈去遍歷原圖得所有 pixed，但這樣做會有個缺點，當圖像越大時，就會跑得越久。

```
for row in range(h):
    for col in range(w):
        Ip_dash[row][col] = self.Bilateral_Filter(row, col)
Ip_dash = Ip_dash.reshape((w, h, 3))
```

改進後的方法：

用 for 迴圈去遍歷 kernal，在計算時，用 numpy 的矩陣運算一次把整張圖算進去，能省去不少時間。

```
for row in range(self.wndw_size):
    for col in range(self.wndw_size):
        # (Ip - Tq) ** 2
        I_diff = (guidance - padded_guidance[row : row + h, col : col + w]) ** 2
        Iq = padded_img[row : row + h, col : col + w]
```