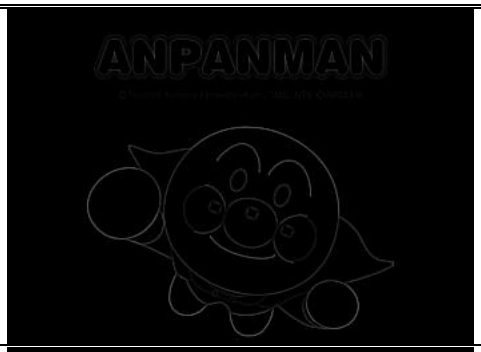
# Computer Vision HW1 Report

Student ID: R12521601

Name: 詹承諺

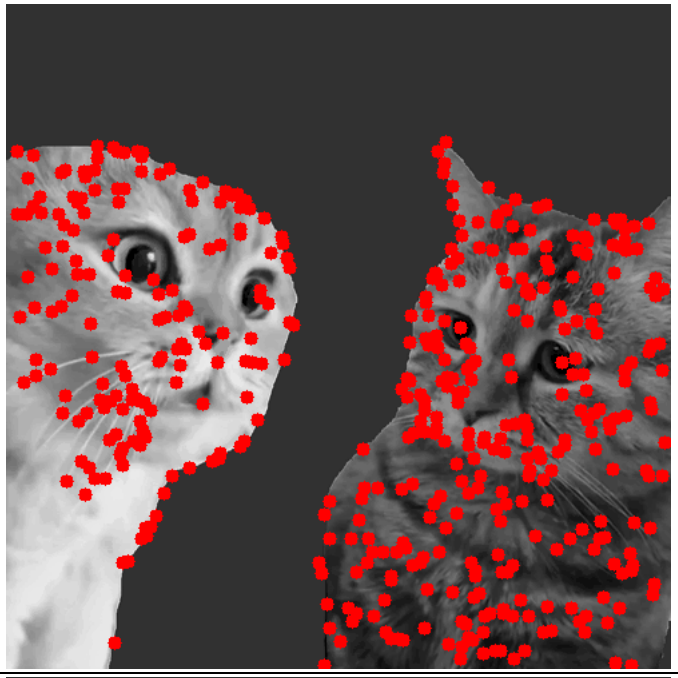
## Part 1.

- **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 |  |

(describe the difference)

隨著 threshold 的提高，keypoints 會變少，更能精確地抓到邊緣。相反地 threshold 較低的話，會導致 keypoints 抓到一些變化不大的地方，使得邊緣的檢測較為粗糙、不精準。

## 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 | 1393620 |
| R*0.1+G*0.4+B*0.5 | 1279697 |
| R*0.8+G*0.2+B*0.0 | 1127913 |

| Gray Scale Setting | Cost (2.png) |
|---|---|
| cv2.COLOR_BGR2GRAY | 183850 |
| R*0.1+G*0.0+B*0.9 | 77882 |
| R*0.2+G*0.0+B*0.8 | 86023 |
| R*0.2+G*0.8+B*0.0 | 188019 |
| R*0.4+G*0.0+B*0.6 | 128341 |
| R*1.0+G*0.0+B*0.0 | 110862 |

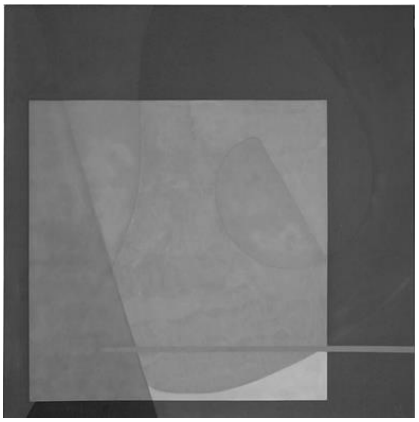
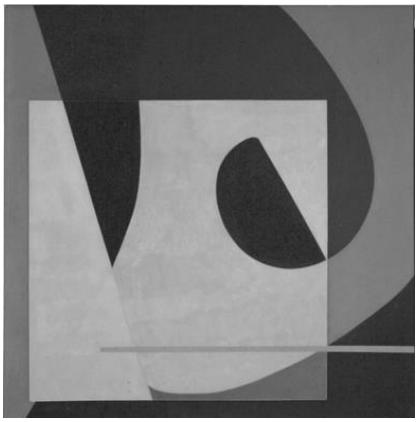
- **Show original RGB image / two filtered RGB images and two grayscale images with highest and lowest cost.**

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

(Describe the difference between those two grayscale images)

cost 大的灰階圖難以看出顏色差異，導致一些原本是顏色差異很大的區塊卻看起來顏色相近。

而 cost 小的灰階圖可見紅色楓葉的部分較白較明顯，相較 cost 大的灰階圖較能看出顏色差異。

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

(Describe the difference between those two grayscale images)

cost 大的灰階圖各種顏色的差異不明顯，如原圖方塊內的橘色和紫色在灰階圖卻看起來顏色相近。

而 cost 小的相較於 cost 大的灰階圖更能看出顏色差異，不論是方塊外的紅色與藍色，或是方塊內的橘色與紫色，都比起 cost 大的灰階圖更能看出色差。

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

```
Gr_LUT = np.exp(np.divide(-(np.arange(256)/255) * (np.arange(256)/255) , (2 * self.sigma_r**2)))
```

利用 look up table 的方式可以節省計算，讓每次計算 range kernel 的時候都能回去查表，之後在算 Gr 的時候就不需要每格都去作計算。