

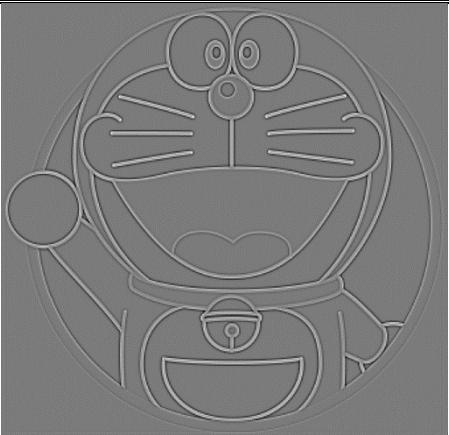

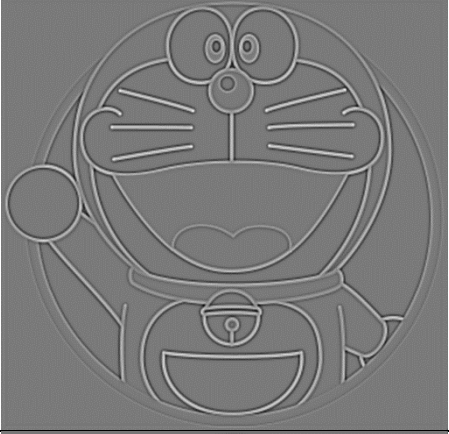

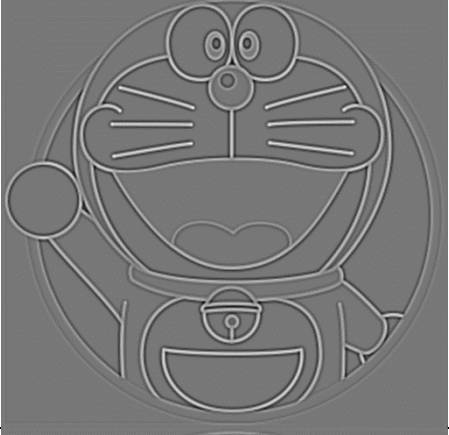

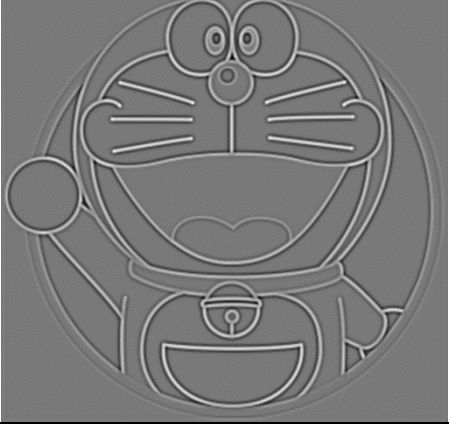

Computer Vision HW1 Report

Student ID: R10921061

Name: 袁肇謙

Part 1.

- Visualize the detected corner for 1.png.

	DoG Image (threshold = 5)		DoG Image (threshold = 5)
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 (2, 5, 7) on 2.png and describe the difference.

Threshold	Image with detected keypoints on 2.png		
2			
5			
7			

(describe the difference)

I observe that the image with larger threshold contains less keypoints. The values that don't reach the threshold won't be considered as keypoints. Some of these small values may be caused by noise. Therefore, the keypoints filtered with larger threshold are more robust.






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




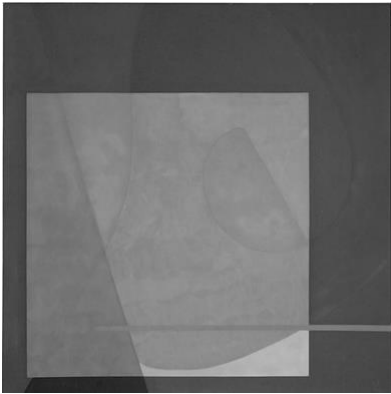
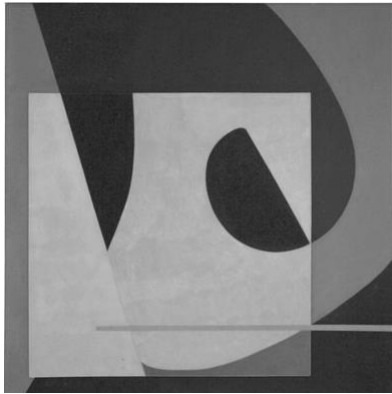
Gray Scale Setting	Cost (2.png)
cv2.COLOR_BGR2GRAY	183850
$R*0.1+G*0.0+B*0.9$	77883
$R*0.2+G*0.0+B*0.8$	86023
$R*0.2+G*0.8+B*0.0$	188019
$R*0.4+G*0.0+B*0.6$	128341
$R*1.0+G*0.0+B*0.0$	110862

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

Original RGB image (1.png)	Filtered <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
	 	 

(Describe the difference between those two grayscale images)

The original image is mostly composed of red and green. Therefore, the grayscale image with the higher proportion of red and green components will have the lower cost. As the result shown above, the gray scale setting $R*0.8+G*0.2+B*0.0$ is pretty close to the color ratio of the original image. The corresponding gray scale image is distinguishable and leads to the lowest cost. On the other hand, the gray scale setting only composed of blue component has the largest cost and the corresponding gray scale image is low contrast.

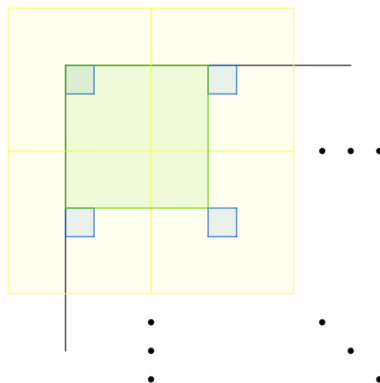
Original RGB image (2.png)	Filtered <u>RGB image</u> and <u>Grayscale image</u> of Highest cost	Filtered <u>RGB image</u> and <u>Grayscale image</u> of Lowest cost
		
		

(Describe the difference between those two grayscale images)

The original image is mostly composed of blue and red. As the result shown above, the gray scale setting $R*0.1+G*0.0+B*0.9$ is pretty close to the color ratio of the original image. The corresponding gray scale image has higher contrast. Other the hand, the gray scale setting mostly composed of green component has the largest cost and result in low contrast gray scale image.

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

For an image of size $H \times W \times C$, the original way to implement a bilateral filter is to calculate the intensity of the filtered image pixel-by-pixel, which takes HW iterations. I speed up the implementation of bilateral filter by building LUTs and calculating all the pixels with non-overlapping windows at once.



The blue squares are the target pixels in the same iteration. The non-overlapping yellow squares are the corresponding windows of each target pixel in the same iteration. After each iteration, I target the pixels beside (right/below) the original ones. My implementation cost window_size^2 (area of green square) iterations to get the intensity of all pixels of the filtered image, which is much less than the brute-force way.