



Computer Vision HW1 Report

Student ID: R13921072
Name: 何家祥

Part 1.

- Visualize the DoG images of 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

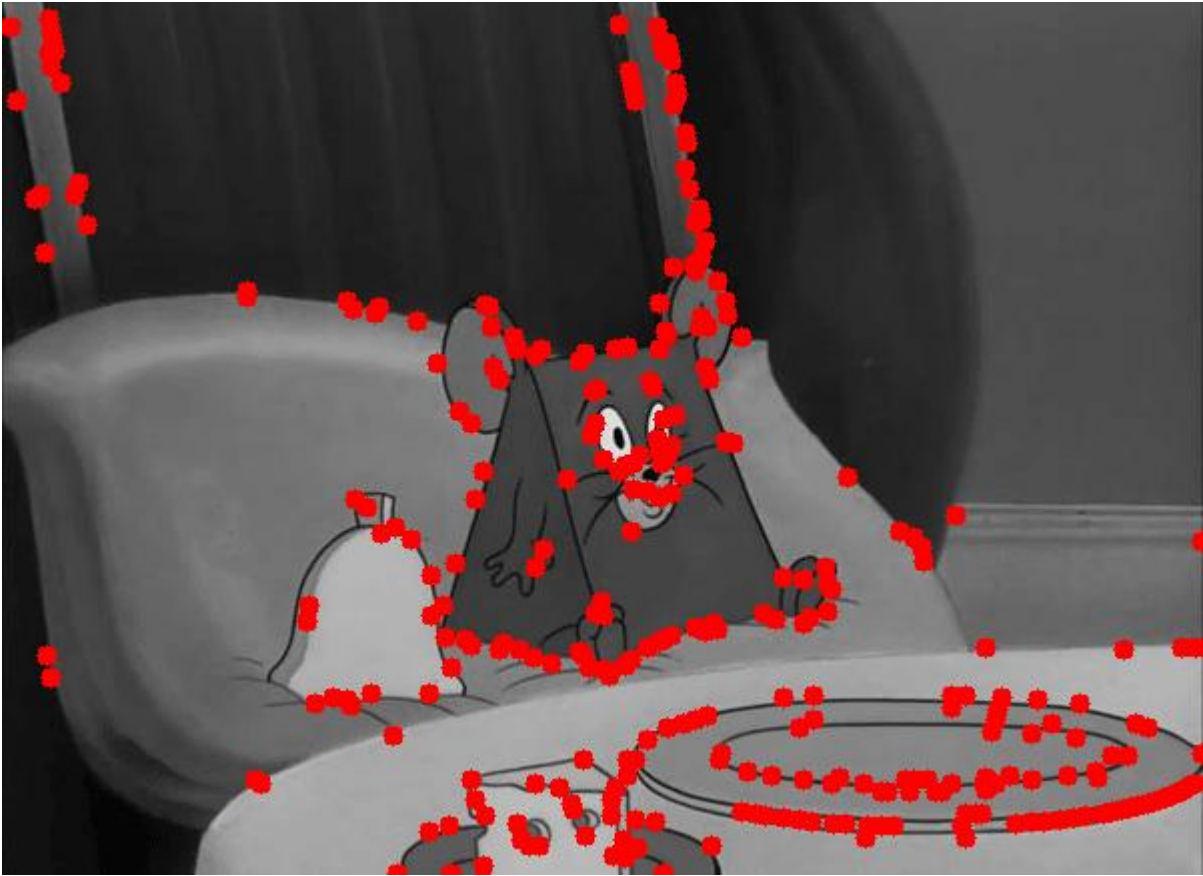


DoG2-
3.png



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)

The distinction among the three PNG pictures lies in the quantity and clarity of the keypoints identified. A lower threshold captures a greater number of keypoints, including those with minimal contrast and noise, whereas a higher threshold focuses solely on the most distinct keypoints with maximum contrast.

When the threshold is set to 7, keypoints are detected only on the Jerry mouse itself and the cheese, where the differences are particularly pronounced, while regular patterns like the plate and sofa no longer show any keypoints.






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	183851
$R*0.1+G*0.0+B*0.9$	77884
$R*0.2+G*0.0+B*0.8$	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.**




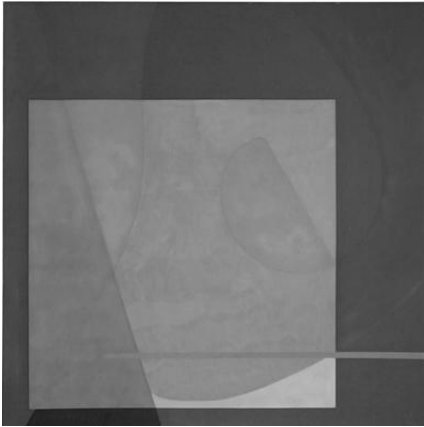
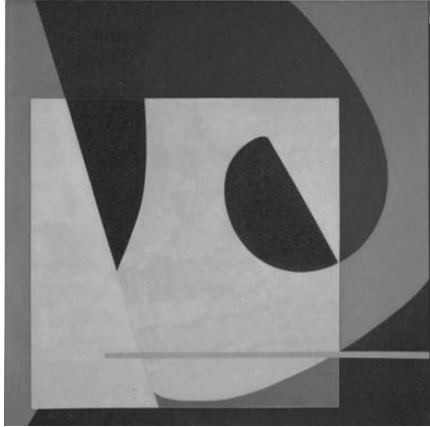
Original RGB image (1.png)	Filtered <u>RGB image</u> and <u>Grayscale image</u> of Highest cost	Filtered <u>RGB image</u> and <u>Grayscale image</u> of Lowest cost
		
		

(Describe the difference between those two grayscale images)

By observing the original image, it is clear that there is no blue at all. This suggests that if the blue channel has a high weight during the linear transformation to grayscale, it may fail to highlight differences in the image, leading to a higher cost.

The actual results confirm this assumption—the two images with the highest cost have blue channel weights of 1.0 and 0.9, respectively. From the results, it is evident that these weights fail to preserve the original color differences.

On the other hand, the image with the lowest cost has a blue channel weight of 0, while the red and green channels—representing the colors of the grass and the flower—are appropriately weighted. As a result, the differences in the image are more distinct, and the edges of the flower are easier to recognize.

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)

In the second image, at first glance, it appears colorful, but upon closer observation, it becomes clear that the dominant visual elements are primarily composed of blue, including the blue background and the purple (R+B) in the central area. Therefore, it can be inferred that the lowest-cost combination should consist of a slight amount of red with a higher weight on blue.

Additionally, JBF considers both color differences (range) and spatial differences, which results in a sharper and more defined outcome. On the other hand, in the highest-cost image, the highest weight is assigned to green, which is actually the least present color in the original image. As a result, it completely fails to accurately reflect the true appearance of the image. The final output looks like two simple square blocks, which is far from the intended effect.

- **Describe how to speed up the implementation of bilateral filter.**
- Use 11th Gen Intel(R) Core(TM) i5-1135G7 40GHz
- My original approach was to brute-force break down the math from the lecture slides, step by step. However, the speed was extremely slow, taking about 3 seconds.
- Later, based on the tips provided by the teaching assistant, I designed a LUT to move the more complex exponential calculations outside the loop. This allowed me to replace the point-by-point heavy

computations with something similar to table lookups, which significantly reduced the execution time, successfully bringing it down to 1.5 to 1.7 seconds.