Biomedical Image Processing – Fourier Domain Filtering

Paul Vu
301169550
March 4th 2017

SFU

Project Overview

In this project I implemented an algorithm that can determine if an image is predominantly low frequency vs predominantly high frequency. The image will be given a score between 0-100 where if a score is > 50 it is a low frequency image and if it's < 50 it is a high frequency image.

MATLAB: FFT2 & FFT Shift

In order to measure an image's frequency, we need to convert the image matrix into the frequency domain. This can be done with MATLAB's FFT2 function which takes an MxN image and calculates the two-dimensional Discrete Fourier Transform. The output from an FFT2 will have the high-frequency components in the center of the matrix, while the low Frequency components are spread to the corners. By using MATLAB's fftshift function we can swap the corner pieces with the center to move low frequency to the center and high frequency to the edges. This will be necessary for filtering.

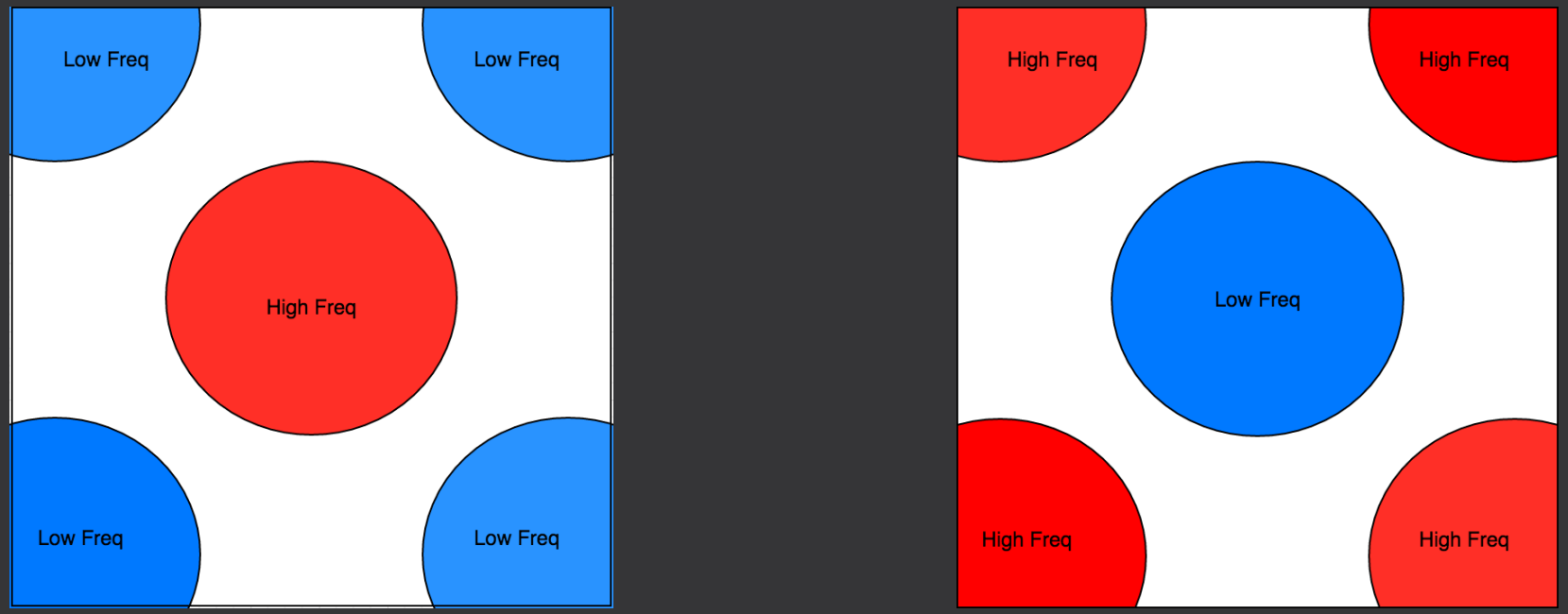

Fig. 1: The left image shows the output of an FFT with high frequency components centered. While the right image shows the result of shifting this FFT output

Low & High Frequency Filter

To filter an image we simply create a circular filter, which will only capture the center values of a frequency matrix. The strategy is to apply this filter to a low-freq centered matrix or high-freq centered matrix, thus filtering out the non desired frequencies. We will then apply the inverse FFT transform to return to our newly filtered image. For design purposes, I selected a circular filter of with diameter of about 10% of the image's width and length. I tried values of various sizes, however the smaller the diameter the better the result. This is because the high frequency values are centered therefore the larger the diameter of the filter, the more lower frequency values filtered out which leads to inaccurate results.

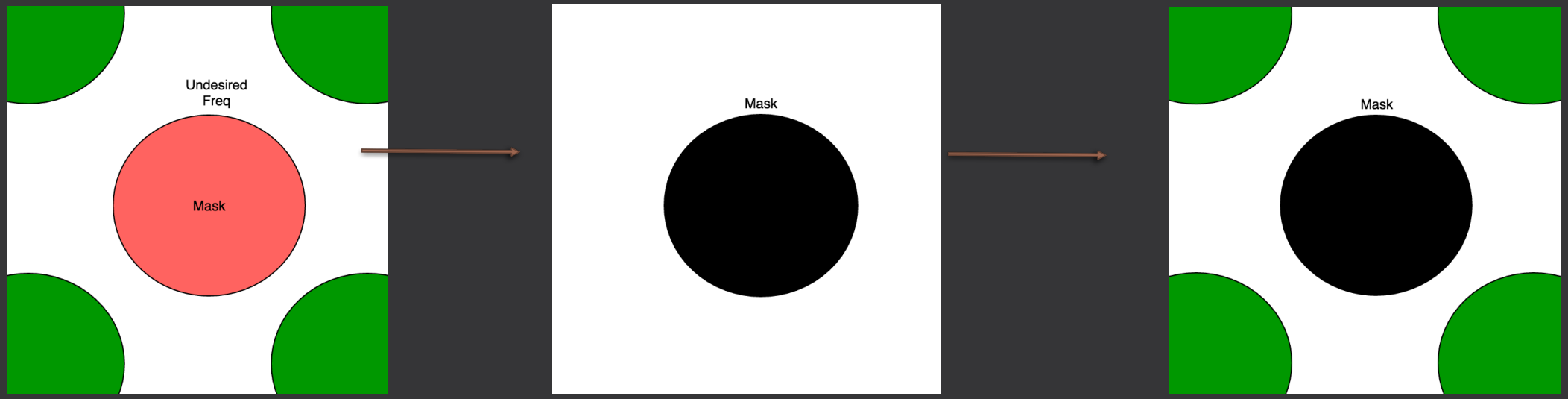

Fig. 2 Frequency Filtering Methodology, applying a mask to filter out the undesired Frequencies

Algorithm Frequency Scoring

Before I shifting the matrix, I have values in the frequency domain with real and imaginary values. I choose to only look at the real components, and sum all the values in the centered circle which contains the High Frequency components. I then divide the sum by the maximum possible value of the sum, which is if the entire circular area was high frequency (value of 1). From there I will have an estimate if the image is high or low frequency. It is important to know that the accuracy of this method will heavily depend on the size of the sampled center circle, for that reason I will only concentrate on a smaller centered circle of 10% length and 10% width of the of the entire image. This will improve accuracies since high frequencies are centered after FFT2.

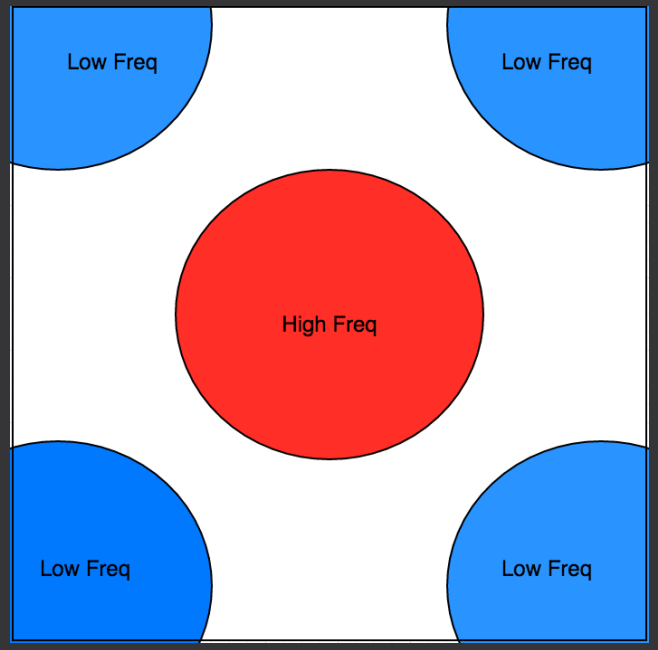

Fig. 3 Scoring will only look at summing the high Freq component

Frequency Filtering Results

When filtering out all HIGH Frequency components, we are essentially removing the sharp edges from the image. This can be shown by subtracting the original image with the filtered image. Thus we can see exactly the components that were filtered out. If we simply display the filtered image, it might be difficult to see the impact of my algorithm if an image has minimal sharp edges.

When filtering out all LOW Frequency components, we are removing all NON-edges from the image, therefore by displaying the filtered image we will see the edges of the image.

Several samples are shown in the next few slides.

Creating Images: Image 1

Using my “freqAnalyzer function”, I take in an image and output a low frequency and high frequency image filtered version of the same image.

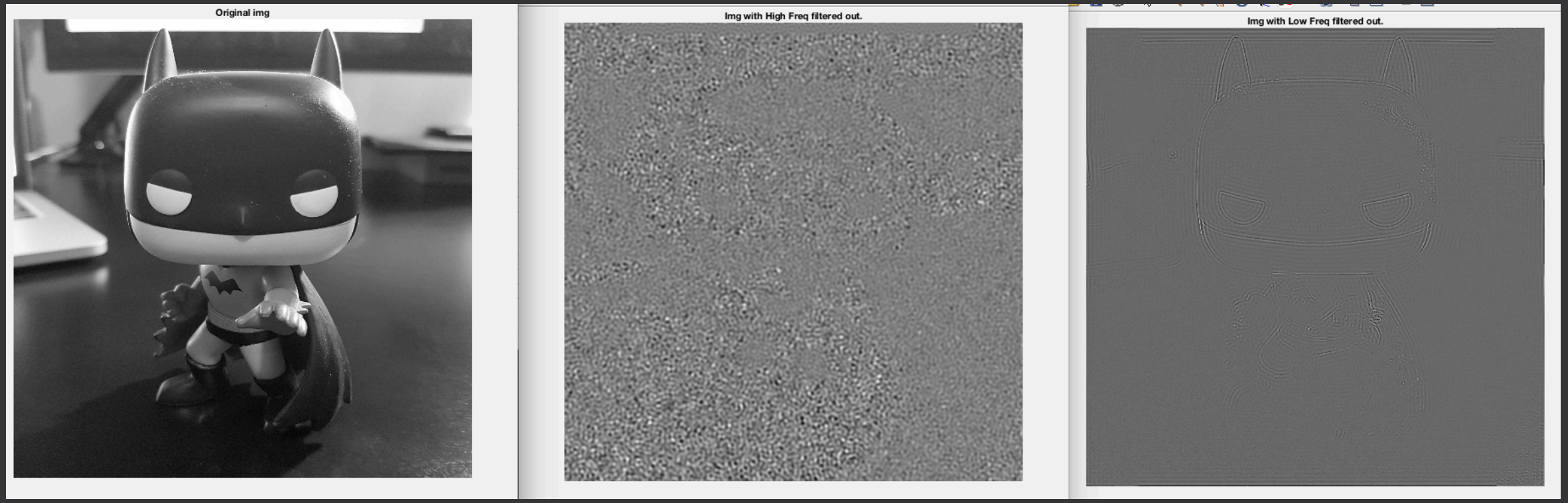

Fig. 4 Image 1 with it’s high and low frequency components filtered

Creating Images: Image 2

Using my “freqAnalyzer function”, I take in an image and output a low frequency and high frequency image filtered version of the same image.

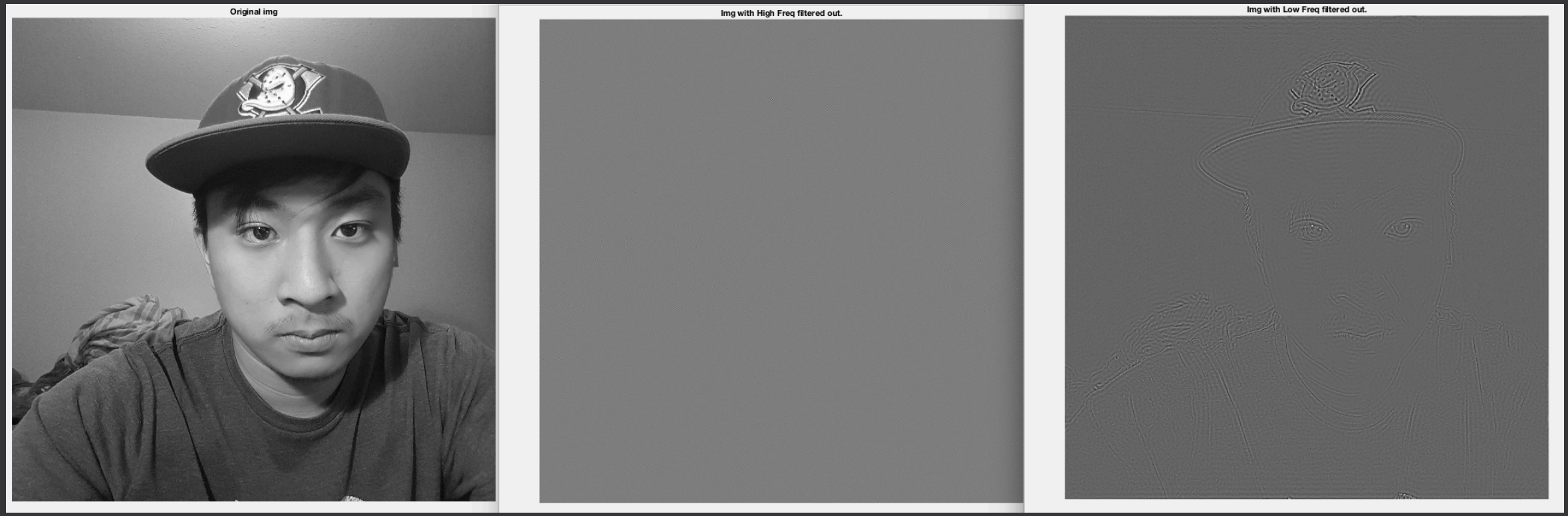

Fig. 5 Image 1 with it’s high and low frequency components filtered

Creating Images: Image 3

Using my “freqAnalyzer function”, I take in an image and output a low frequency and high frequency image filtered version of the same image.

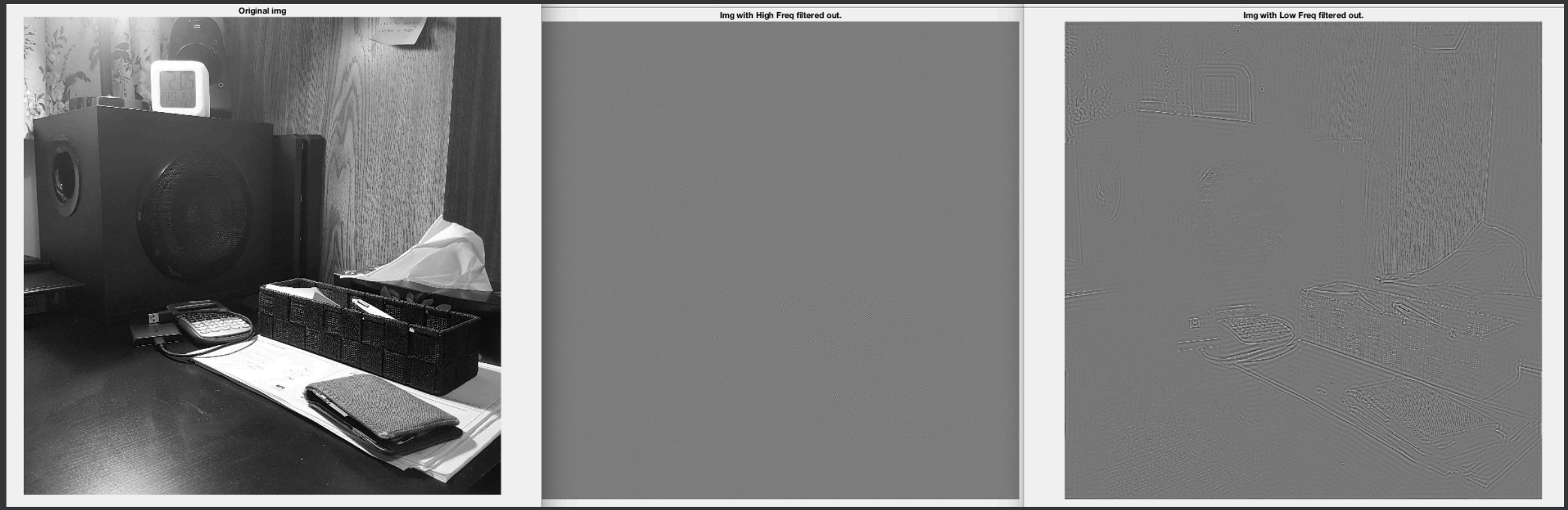

Fig. 6 Image 1 with it’s high and low frequency components filtered

Creating Images: Image 4

Using my “freqAnalyzer function”, I take in an image and output a low frequency and high frequency image filtered version of the same image.

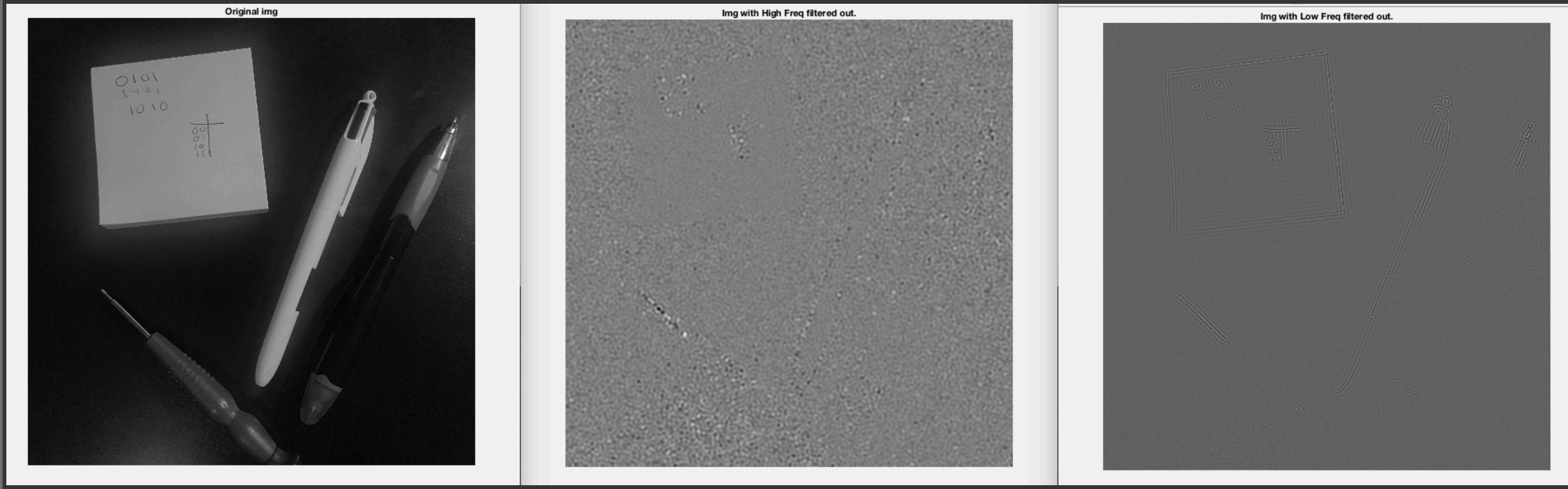

Fig. 7 Image 1 with it’s high and low frequency components filtered

Score Results

In all cases the original image shows the highest score. This is intuitive since the other images are filtered versions of the original image, thus some high frequency components were filtered out.

My algorithm is proven to work since for images with filtered high frequency values (meaning high freq was removed) gives an output of near 0 value. While images with low freq components filtered out give larger values typically >40. In all cases these images are “low” frequency images since neither oar aboe 50, but the results do show that the filtered low freq image has MORE high freq components and thus a higher score.

Batman Img	Image:	Score:		Selfie Img	Image:	Score:
	original_img	84.0363			original_img	90.96
	filtered_high_freq_img	0.00037			filtered_high_freq_img	0
	filtered_low_freq_img	42.2			filtered_low_freq_img	47.22
Pens &Notes Img	Image:	Score:		Speakers Img	Image:	Score:
	original_img	72.97			original_img	84.57
	filtered_high_freq_img	0.01			filtered_high_freq_img	0
	filtered_low_freq_img	49.88			filtered_low_freq_img	49.9

Fig. 8 Results of High Frequency Scoring algorithm