Assignment 4
Deadline: Nov 28th, 11:59 pm

Q1. a. Find W. Consider L to be the FFT of Laplacian. G is the observed image, F is the true image. All other variables have their standard meanings.

$\sum_k \sum_l |W(k,l)G(k,l) - F(k,l)|^2 + \gamma \sum_k \sum_l |L(k,l)W(k,l)G(k,l)|^2$ [1]
Evaluation will be done during demo.

b. Use the above filter to denoise a corrupted image.
- Take Barbara image and add AWGN of mean 0 and variance of the order of 100. Note you can use any inbuilt function for this. Sometimes they may scale the values and may be expecting values of variance in the order of 0.01 or even smaller. In whatever way you give noise variance, make sure that noise is visible to human eye.
- Since F, N are unknown, replace them with a constant, say K. Using a grid search, check for what value of constant K and γ the denoised image gives best PSNR. For G, consider using $|G|^2 = |HF|^2 + |N|^2$
- Show the original image, noisy image, denoised image with best PSNR [2]
- Compare the results with Wiener filter, that is, $\gamma = 0$

Q2. a. Take the noisy image from previous question. Apply a Log filter with $\sigma = 2$. Note you must obey the filter size based on σ. You can use a library to obtain LoG filter. You can use inbuilt convolution to compute the responses.
Perform zero-crossing using a 4% threshold. You must do it without using library.
Show the clean image, noisy image, filter, response after convolution, edges after zero-crossing. [2]
b. Now obtain the horizontal and vertical 1-D Gaussian filters using $\sigma = 2$. Apply them on the noisy image. You can use inbuilt 1-D convolution. If that is not clear, you can use 2D convolution. Display the response. [1]
Use horizontal and vertical second order derivative filters and apply them over the obtained response. Show both responses. [1]
Now add these responses and compare with the LoG response in (a). You can compute difference and then take sum of absolute of difference. Display this number. Display the difference by converting it into [0, 255]. You can first subtract the min, and divide the resultant by max, then multiply by 255. [1]
Visually the difference should not be too significant.
Perform zero-crossing using a 4% threshold. You must do it without using library.

Show the clean image, noisy image, 1-D Gaussian filters, response after Gaussian filtering, response after second order derivative filtering, edges after zero-crossing.
If you are able to use 1D convolution, you will see a clear difference in the time involved in parts (a) and (b).