Lab 5 Fourier Space, ECE 237 Spring 2021

1. Inspect the 2D Fourier transform (real and complex/imaginary part of the FFT correspond to the amplitude and phase of the light field that forms the image) and the Fourier power spectrum (which is the square of the absolute value of the FFT and which corresponds to the light intensity of the image) of a couple of images:

    a. In MATLAB, use the code below to open an image of your choice and inspect the real and imaginary part of its Fourier transform:
    Remember your complex math: the imaginary number y = x + $\boldsymbol{i} \cdot$ y consists of the real part x and the imaginary part y, and $\boldsymbol{i}$ is the imaginary unit (defined as $\boldsymbol{i}^2 = -1$)

    ```
    M = imread('Flamingo.jpg');
    Mbw=rgb2gray(M);
    imagesc(fftshift(log(imag(fft2(Mbw)).^2)))  % imaginary part
    imagesc(fftshift(log(real(fft2(Mbw)).^2)))    % real part
    ```

    Note:
    The *fftshift* function displays the data more intuitively for us, with the zeroth order in the center, just like we would see in physical reality if we inspected the light field in the back focal plane of a lens. MATLAB standard notation otherwise puts the zeroth diffractive order in the upper left corner.
    We use log before plotting in order to display the data in log scale so that we can see the information better.

    b. Display the Fourier power spectrum of the image. This is the square of the absolute value of your Fourier transform.

2. Apply some spatial frequency filters in Fourier space. First, transform your image to Fourier (frequency) space using the fft2() command. Then, apply the filter (remember, in Fourier space we simply multiply our filter with the image, instead of performing the convolution operation as we do in real space filtering).

    a. Implement a high spatial frequency filter with a binary mask.
    b. Implement a low spatial frequency with a binary mask.
    c. Implement a band-pass spatial frequency filter with a binary mask.
    d. Now do the same filtering with a smooth transition “non-ideal” mask (e.g. Butterworth) instead of the “ideal” binary mask with a sharp edge. We do this to reduce the ringing artifacts caused by the sharp edge of the mask. Can you see the difference?
    Test how your different filters perform on a few different images that you like and test different cutoff frequencies for the filters.

OPTIONAL EXTRA MATERIAL FOR MORE EXPORATION (will not be graded)
Explore padding the image before applying the Fourier transform to get rid of ringing artifacts from the edges. (Remember to crop your result to the original size after you transform it back to real space.)
Explore how a convolution kernel (such as a Gaussian kernel) corresponds to a Fourier space frequency mask by doing the fft2() of the kernel. Compare the result of convolution in real space with your kernel to the result of multiplying the Fourier transform of the kernel and the image! If you do this properly they will be identical (except for some truncation errors)
Create your own Laplacian of Gaussian (LoG) filter and try this out on the images.