Experiment 4.2

Frequency domain filtering

Q1. Perform the following frequency domain filtering by writing your own Python functions (LPF= Low Pass Filter, HPF = High Pass Filter).

a. Ideal LPF, Ideal HPF

b. Gaussian LPF, Gaussian HPF

c. Butterworth LPF, Butterworth HPF

INPUT: Image filename and cut off frequency are the input arguments.

OUTPUT: Display the (shifted) magnitude spectrums of the input, of the filter and the of filtered output. You may use the tracker/slider function to choose 1) images, 2) filter types and 3) cut-off frequencies.

Q2. Read the "leopard_elephant.jpg" image. This is an example of an image illusion in which the perception of an image changes with viewing angle, time spent viewing, or image size. A leopard can be seen in the image if you view it at its full spatial resolution. However, if you view the image at a lower spatial resolution, an elephant will appear. Using your frequency domain filtering idea, can you create an identical optical illusion. Read the files "einstein.png" and "marilyn.png." Make a hybrid image so that when viewed at a higher spatial resolution, the illusion appears to be Einstein, and when viewed at a lower spatial resolution, it appears to be Marilyn.

INPUT: einstein.png and marilyn.png.

OUTPUT: Hybrid image formed by einstein.png and marilyn.png

Q3. Read the "cameraman_noisy1.jpg" image. What kind of distortion did this image undergo? Could you provide some insight into the physical events that caused this distortion? Create a function that will automatically remove noise from images of this kind. Check your algorithm's robustness by seeing how well it can eliminate noise from the "cameraman_noisy2.jpg" image.

INPUT: cameraman_noisy1.jpg, cameraman_noisy2.jpg.

OUTPUT: Respective filtered image.

NOTE:

1. Do not hardcode the filenames and/or image size into the code.

2. Use proper code commenting and documentation.

3. Use self-explanatory identifiers for variables/functions etc.