NOVA SCHOOL OF SCIENCE & TECHNOLOGY

Advanced topics in Digital Image Processing

AULA 8 – Processing in the Frequency Domain and Image Debluring

Objective:

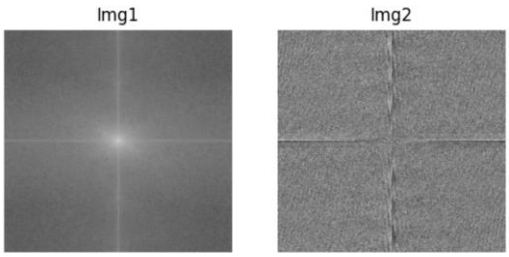
In this class image processing algorithms in the frequency domain, using the fast Fourier transform (FFT), should be experimented.

The exchange of phases and amplitudes between different images will be tested and applied low-pass and high pass filters in the frequency domain.

Procedure:

0. **Preparation** – Add image*FFT.py and imageDeconvolution.py* files to the project.

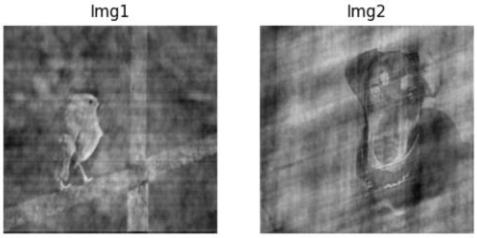
1. **Visualization** – Apply the Fourier transform to an natural image and visualize its amplitude spectrum (in logarithmic scale) and phase.



Commands to use:

- `GetFFT_Mag_Phase`
- `cv.log`
- `np.fft.fftshift`

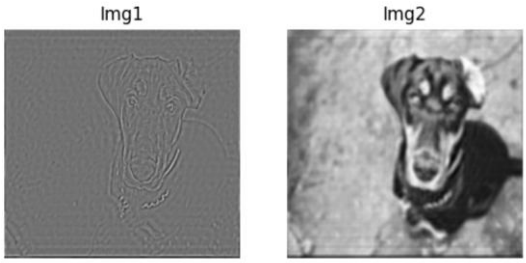
2. **Exchange of amplitude and phase** – Open images, aula8 (1) and aula8 (2) and apply the Fourier transform. Then perform the inverse transform replacing the phase and amplitude of each one of the images and view the resulting images.



Commands to use:

- `GetFFT_Mag_Phase`
- `GetFFT_Inverse_Mag_Phase`

3. **Ideal Filter**: Create a low-pass filter and a high-pass filter with masks in the amplitude and apply to an image. The cutoff frequency for the filter should be asked to the user (e.g. between 20 and 50).



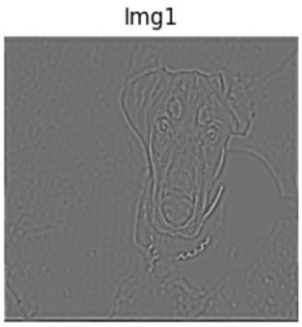
Ideal Low-pass Filter

$$H(u,v) = \begin{cases} 1 & D(u,v) >= fc \\ 0 & D(u,v) < fc \end{cases}$$

Commands to use:

- `GetFFT_Mag_Phase`
- `GetFFT_Inverse_Mag_Phase`
- `GenerateFilterMask_Ideal (ideal filter)`
- `Cv.multiply`

4. **Gaussian Filter**: Use a Butterworth filter and compare the results with the previous ones. You should see a decrease in the ringing effect The cutoff frequency for the filter should be asked to the user (e.g. between 20 and 40).



Commands to use:

- `GetFFT_Mag_Phase`
- `GetFFT_Inverse_Mag_Phase`
- `GenerateFilterMask_Gaussian`
- `Cv.multiply`

Debluring

5. Using image “aula5 (3)”, blur the image either using a Gaussian filter and a Motion Blur filter.

```
imgResGauss = cv.filter2D(imgOriginal,cv.CV_8U, filter,
anchor=(np.int32(filter.shape[0]/2), np.int32(filter.shape[1]/2)))
```

Commands to use:

- `-GetFilterConv`
  `-cv.filter2D`

6. Add white noise to the image. Ask the user the intensity of noise (e.g. 10).

```
imgRand = np.random.rand(imgOriginal.shape[0], imgOriginal.shape[1]) * noiseAmp
```

7. Test the image debluring method by Wiener filtering with Gaussian filter, with and without noise and with different K values (*Signal to Noise Ratio*) (0.0001 … 0.1)..

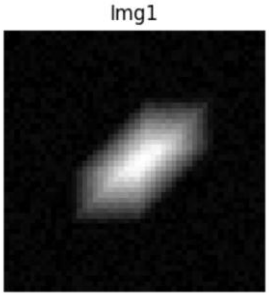
$$\widehat{F}(u,v) = \left[\frac{H(u,v)^*}{|H(u,v)|^2 + K} \right] \cdot G(u,v)$$

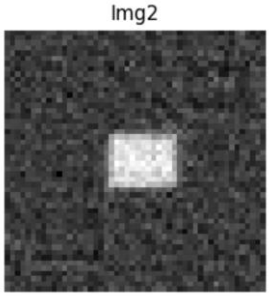
Recovered with (snr = 0.001 + noise= 0 & cutoff = 1)

Commands to use:

- `InverseDeconvolutionWiener`
-

8. Test the image debluring method by Wiener filtering to recover shaken images. Use the motion blurred image from 5.





Recovered with (snr = 0.001+ noise= 10)