IMPR 2017

Exercise #4

Due date: Jan. 11, 2017, 11:55pm.

This exercise can be submitted in pairs.

The goal of this exercise is to practice the basic concepts of the Fourier transform. It includes both theoretical part and a programming part. In the theoretical part you will practice hands-on calculation of the Fourier transform on 1D signals, along with proving some of its basic properties. In the programming part you will implement a simple 1D version of the Fourier transform and use frequency domain operations to practice some image processing techniques in the frequency domain.

You are not allowed to use opencv built in functions for the tasks in this exercise except reading images from disk. In task 2 you are not allowed to use any existing implementation of the Fourier transform. In parts 3-5 you are allowed to use the numpy fft implementation.

Task 1: Fourier transform - Theoretical part

a) Calculate by hand the 1D discrete Fourier transform $F(k)$ of the signal:

$\mathrm{x_k} = [\,2, 1, 3, 1, 3, 2\,]$. You need to show the detailed calculation and not just the final result.

b) Calculate by hand the original signal x_k given its Fourier transform:

$$F(k) = [12,\ -0.5-0.866i,\ 1.5+0.866i,\ 4,\ 1.5-0.866i,\ -0.5+0.866i]$$

c) Show that the Fourier transform $G(w)$ of a sine function with wavelength ω_0 :
$g(x) = 2\pi\omega_0 x$ is: $\frac{i}{2}\Big(\big(\delta(\omega+\omega_0)\big) - \delta(\omega-\omega_0)\big)\Big)$

d) The Discrete Fourier Transform (DFT) is defined as: $F\big(k\big) = \sum_{n=0}^{N-1} x_n e^{\frac{-i2\pi k}{N}}$.

Show that the DFT is cyclic, i.e.: $\mathrm{F(k+N) = F(k)}$

Task 2: Implement a direct and inverse 1D Fourier transform

a. Implement a function that gets as input a 1D signal of arbitrary length and return its Fourier transform (of the same length of the signal) in Cartesian format (i.e. a+bi).

b. Implement a function that gets as input a 1D Fourier transform of some signal in Cartesian format (i.e. a+bi) and return the original signal.

c. Implement a function that gets as input a 1D signal of arbitrary length and return its Fourier transform (of the same length of the signal) in Polar format (i.e. $Re^{i\theta}$).

d. Implement a function that gets as input a 1D Fourier transform of some signal in Polar format (i.e. $Re^{i\theta}$) and return the original signal.

Relevant numpy functions: numpy.arctan2 (https://docs.scipy.org/doc/numpy-1.13.0/reference/generated/numpy.arctan2.html#numpy.arctan2)

Note that you are not allowed to use any built-in functions to perform the conversion between Cartesian and polar coordinates.

Task 3: Image upsampling in the frequency domain

Implement a 2D image upsampling function. The algorithm should be the zero-padding in the frequency domain. The function should get as input a 2D image and a 2D vector with the upscaling parameters (larger than 1) at each dimension. The function should return the following: The FFT of the original images, the zero-padded FFT, and the upsampled image. You may use numpy fft functions: fft2, ifft2, fftshift, ifftshift, in your implementation.

In case of scaling factor < 1, the function should return the original image and its Fourier transform for both the Fourier and the zero-padded Fourier return values.

Task 4: Image translation alignment in the frequency domain

In this task you should implement an algorithm that finds a translation between two images sampled from an original image. You should implement the phase-correlation algorithm which use the frequency domain to find the translation in x, y between two images. The algorithm described at: https://en.wikipedia.org/wiki/Phase_correlation#Method

You can ignore the first step (application of Hamming window) in your implementation. You do not need to consider sub-pixel translations or any rotations.

Task 5: Image filtering in the frequency domain

a. In this task you should implement a band-pass filtering function, that allow both high-pass, low-pass and band-pass filtering in the frequency domain. The function should get as input the original image, the low-pass threshold and the high-pass threshold. In case of low-pass filtering, the low-pass threshold should be 0, and in case of high-pass filtering the high-pass threshold should be larger than the largest possible wavelength in the image.

b. Create a series of filtered images, with various low-, band- and high-pass values and print them into a pdf document that should be submitted as part of your solution. Each filtered image should include a title with the parameters of the filter applied to this image.

c. Applying low-pass filtering using the function you wrote will result in ringing artifact. Explain in your readme what is the source of these rings and which filter can be used for low-pass filtering without ringing artifacts. You should justify theoretically why the proposed filter will not suffer from ringing artifacts.

d. Image deconvolution: in this task you are requested to implement the Weiner filter for image restoration as described in class. The provided driver demonstrates the application of the deconvolution process on an image degraded with motion blur kernel and with and echo degradation kernel. Your function should get as input the degraded image, the convolution kernel used to degrade the image and the lambda parameter to avoid dividing by zero. The function should return the restored image.

Files included:

Ex4driver.py: an example that defines and demonstrates the API for the functions you should implement.

ex4.pyc: binary version of school solution, so you can run and get an idea what your solution should look like.

./Images/* :set of images to be used with the driver

You should submit:

Ex4.zip file contains:

1. ex4.py: the code of your solution. The file ex3driver.py should run with your ex3.py without any error or warning.

2. ex4.pdf: a **pdf** file contains the answers to task 1 and the figures you should generate in tasks 5b, 5c. **Note: we will not accept documents in any other format. Only PDF is acceptable.**

3. **<u>README.txt</u>**: include your names in the first lines. **Note: any deviation from the format described here will incur a penalty.**

<u>Submission instructions:</u>

Through the course moodle website