Lecture "Digital Signal Processing"

Prof. Dr. Dietrich Klakow, Summer Term 2021

Assignment 2

Submission deadline: 03 May 2021, 23:59

Submission Instructions:

You have one week to solve the assignments.
The code should be well structured and commented. Do not use any Matlab-Toolbox or Python external libraries if it is not mentioned that you can use them.

- You are allowed and encouraged to hand in your solutions in a group of two students.
- There are two parts in this assignment: theoretical part and practical part.
- The practical part that is solved with Python should be submitted as an ipynb file (Jupyter Notebook or Google Colab), where every function is written in a separate block.
- The theoretical part can either be written by hand and scanned, or typed with LaTeX in the text / markdown area of ipynb file.
- Submission of both parts should be done via Microsoft Teams by one of the group members.
- Submission should be named as: Ex02_matriculationnumber1_matriculationnumber2.zip

The submission should contain the following files:

- file "README" that contains an information on all team members: name, matriculation number, and email address.
- code files
- file "answers.pdf" which contains answers to the questions appearing in the exercise sheet. *Note: If you use ipynb file, you don't have to submit "answers.pdf". You can embed your scanned copy or write your answers in the text / markdown area.*

1 Exercise

The goal of this exercise is to implement a gaussian filter for (grey level) images.

1.1 (1.5P) Subtask

The boundary of an image is a problem for a gaussian filter. We will solve this problem by mirroring the image at the boundary.
We will assume odd, square convolution matrices K of size $h \times h$. How far must the image be mirrored if you use such a kernel?
Implement the function *imgmirror* that mirrors a strip of width w at the bounds of the matrix $M \in \mathbb{R}^{n \times m}$. You should get an output matrix $O \in \mathbb{R}^{(n+2 \cdot w) \times (m+2 \cdot w)}$.

Hint: you could use floor function here either in Matlab or Python

1.2 (1.5P) Subtask

Implement the function *gaussfilter*, which filters the matrix $I_1 \in \mathbb{R}^{n \times m}$ with kernel $K \in \mathbb{K}^{h \times h}$ (with standard deviation of σ and zero of μ). Use your function *imgmirror* to handle the boundaries. The matrix $I_2 \in \mathbb{R}^{n \times m}$ is the output of your function.

$$I_2(x, y) = \sum_{i=1}^{h} \sum_{j=1}^{h} I_1(x + i - a, y + j - a) K(i, j) \tag{1}$$

where $a = \lceil h/2 \rceil$.

What is the running time of your implementation, depending on K and I_1?

1.3 (1.5P) Subtask

Use the functions you implemented to denoise the corrupted picture *noisycoke.jpg*. Use a 5x5 smoothing kernel as discussed in the lecture. Vary parameter *sigma*. What do you observe? What is changed?

1.4 (2.5P) Subtask

Now, use the separability property that we mentioned in the lecture to implement a new version of the function $gaussfilter$ .What is the running time of the new version? Compared to subtask 1.2, explain why it is faster?

2 Exercise

The operation we performed in task 1.2 (applying a kernel/window function to each overlapping segment of an image/wave) is also called convolution. In practice, the computation of convolution in time domain will be a problem (quadratic computational complexity). One can show that the convolution can be done easily in frequency domain.
Considering the continuous Fourier transformation:

$$f(x) \xrightarrow{F} \mathcal{F}[f] = \int_{-\infty}^{+\infty} f(x) e^{-i\omega x} dx \tag{2}$$

where ω is frequency. Prove the properties below:

2.1 (0.5P) Spatial shift:

$$\mathcal{F}[f(x-a)](\omega) = e^{-i\omega a} \cdot \mathcal{F}[f](\omega)$$

2.2 (0.5P) Convolution:

$$\mathcal{F}[(K * f)(x)](\omega) = \mathcal{F}[K](\omega) \cdot \mathcal{F}[f](\omega)$$

2.3 (0.5P) Derivative:

$$\mathcal{F}[\tfrac{\partial f(x)}{\partial x}](\omega) = i\omega \cdot \mathcal{F}[f](\omega)$$

$\star$ the properties also hold for discrete signals; using continuous signal here is just for convenience.

3 Exercise

The goal of this Exercise is to give a simple example of building a first and second order derivative filter in 1-D.

3.1 (1.5P) Derivative Filter:

Considering the first order derivative of discrete signal in 1-D case as below:

$$f'[n] = \frac{f[n]-f[n-1]}{\tau} \Longleftrightarrow \underbrace{\frac{1}{\tau}\boxed{\text{-1} \mid 1}}_{convolution\,kernel} * \boxed{\ldots \mid \text{f[n-2]} \mid \text{f[n-1]} \mid \text{f[n]}} \tag{3}$$

where $*$ is convolution and τ is the time changing from f[n-1] to f[n].

Is it a low pass filter, band pass filter or high pass filter?
Give an example of the convolution kernel (of size 3×3) for approximating the second order derivative in 2-D following the same definition above. Where could we use this kind of filter?

Hint: Laplacian of Gaussian