Introduction to Image Analysis

Compulsory Assignment 2

Allan Rasmusson

2023.10.26

Assignment Description

This document (7 pages in total) describes the second compulsory assignment for the course "Introduction to Image Analysis" taking place at Faculty of Mathematics and Informatics, Vilnius University, Lithuania, Autumn 2023. This report is the second of a total of three assignments, it will be scored between 0 and 100 points and weighted by 30% in the overall grade for the course.

The assignment concerns the practical exercises in weeks 6 to 8 of the course, and, as such, can be understood as giving a summary of the material covered in the course so far, a practical implementation of it with some specific tasks to solve using it.

The student must include a report with a short description of the theory behind each relevant topic, what practical challenges it may present and how these and other choices affected the implementation. This must be done for *all* tasks listed in the report, although it is not mentioned explicitly for each task.

Additionally, the student must consider which illustrations (both diagrams and outputs from the implementation) to include to best support explanations and proofs-of-concept. In other words: the report must in itself contain the necessary illustrations to argue that your explanations/example are correct *without* relying on readers running the code.

Both the report *and* the implemented programs (C++ or Python) must be handed in. You must include a description of how each example can be run on a command line in Ubuntu Linux. For the C++ code you must also describe how to compile it using a combination of `make` and `qmake` as well.

It is perfectly fine to hand in multiple pieces for code different examples or solutions to the exercises, and it's also acceptable that the examples must be compiled separately using `qmake` and/or `make`. Examples needing alterations (e.g. commenting out certain sections) of the handed-in code should not be expected to be run and may thus contribute a lower score.

Although the assignment covers the practical exercises already handed out, some clarification and specific tasks are described in the following sections. Some specific data is included with this assignments (`ImgSetCa2.zip`), but feel free to support you explanations using any other data handed out during the course under "Files" in the "General" channel in the MIF-IA Teams Channel (link).

The report and code will contribute equally to the final score and must be handed in a single zip-file uploaded in the VU Emokymai system.[1].

Deadline is Tuesday ~~2023.11.21~~; moved to Tuesday 2023.11.*28*, at 23.59 (1 minute to midnight).

[1]Remember to accept submission of an uploaded draft

Advice

- Ask questions as early as possible.
- If you are having problems with some parts of the exercises, continue to the next and make it work with what you've got.
- Think about which part of the material is relevant for each task.
- Think about which examples and illustrations you need to support your hypotheses/explanations.
- Read the assignment *at least* twice.

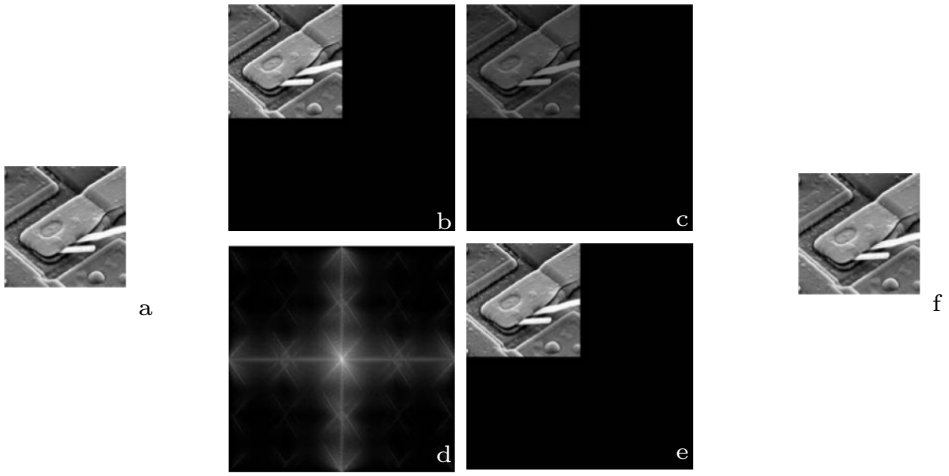
1 Assignment Details

1.1 Fourier Transform

Purpose: Illustrate sufficient understanding of the Fourier transform to implement the necessary steps for performing a Fourier Transform given an implementation of the Discrete Fourier Transform algorithm. C++ implementations will use the `fftw3` library and Python implementations will use the `numpy.fft` as explained in the exercises for weeks 6–8.

Implementations and Tasks:

- Implement the following steps needed to perform the forwards and backwards Fourier transforms of a single image without any additional alterations:
  - a: input image $M \times N$.
  - b: padded image $2M \times 2N$.
  - c: b shifted for periodicity
    by multiplying by $(-1)^{x+y}$.
  - d: **DFT**, $F(u, v)$ of c.
    - Traditionally, filter $(F(u, v)H(u, v))$ is performed here.
    - Instead, create visualization of F for now.
  - e: **IDFT** of d shifted for periodicity
    real part multiplied by $(-1)^{x+y}$.
  - f: after image is upper left quadrant of e.
  - NB: Some of the steps will have to pay attention to **DFT** scaling factor.
  - Without filter application ($H(u, v) = I$) steps e and f will show if the **DFT** and **IDFT** are applied correctly.



Visualization: Argue that the steps have been correctly implemented.

Data: Images handed out during the course.

1.2 Fourier Transform of generated images

Purpose: Building some intuition for what the result of the Fourier Transform represents using generated images.

Implementations and Tasks:

Generate images of various types and *explain* what you expect from the corresponding Fourier spectrum. Show and discuss if the results fit with your expectations.

To generate images of repeated patterns in image space you *may* take *inspiration* in the following (pseudo) code:

```
width = 256;
height = 256;
MyImage im = allocate new  MyImage of size widt x height;

float alphaX = 2.0f*M_PI / width
float alphaY = 2.0f*M_PI / height

for( yIter = 0; yIter < height; yIter++){
  for( xIter = 0; xIter < width; xIter++){
    float i = cos( ``some combination of alphaX|Y and x|yIter'')
    MyImage.setPixel(xIter,yIter, i);
  }
}
```

NB: This is *not* a complete example: you must consider choice of format for `MyImage` and number of intensity levels, range etc.

Consider also if you could gain any insights by generating images in frequency space.

Examples: Generate 3-4 examples, describe your expectations and the outcome.

1.3 Filtering in Frequency Space

Purpose: Understanding/explaining how filtering works in frequency space as opposed to filtering in image space and some of the benefits/drawbacks of filtering in each space.

Implementations and Tasks:

- I: Implement high-pass and low-pass versions of the *Ideal*, *Butterworth* and *Gaussian* frequency space filters and give examples of each filter.
  Extend the implementation in 1.1 to apply the filtering.
- II: Application (in image space) of any linear filter w will yield the same result if w is padded with 0's to be the same size as the input image. Use this and any of the previous implementations to show benefits/drawbacks of:
  - Blurring with an averaging filter in image space vs. frequency space.
  - Blurring with a Gaussian filter in image space vs. frequency space.
  - Differentiation in image space vs. frequency space.
  - Illustrate (and explain) why filtering by Ideal frequency filters leads to ringing artifacts.

Visualizations: Any number you consider relevant for the tasks above. You may use any image from the weekly exercises and this assignment.

1.4 Image Restoration - Noise Removal

Purpose: Gain practical experience with:

- The image degradation/restoration model.
- Basic noise identification.
- Noise removal in image space.
- Noise removal in frequency space.

Implementations

- Filtering in image space (reuse from Compulsory Assignment 1).
- Adaptive filters.
- Band-pass and reject filters.
- Notch filters based on frequency filters from Section 1.3.
- Error measures (also called Objective fidelity criteria).

Tasks:

- I: Periodic noise
  In the data accompanying this assignment is the image `Kidney1-Crop.tif` (original), and two noisy versions of it: `Kidney1-Crop-Noise1.tif`, `Kidney1-Crop-Noise2.tif`
  - Remove the periodic noise from `Kidney1-Crop-Noise1.tif` using band filters and notch filters. Discuss if band or notch is best or if it doesn't matter.
  - Use the same approach to try and remove periodic noise from `Kidney1-Crop-Noise2.tif`. Explain why or why not this approach is successful.
- II: In the data accompanying this assignment is the image `Thorax.tif` (original), and two noisy versions of it: `Thorax-Noise1.tif`, `Thorax-Noise2.tif`
  - Remove noise as you believe is best possible with the methods covered. Describe the steps you take, and the choices you make and how you evaluate your results.

Data: Images included with Compulsory Assignment 2.