TDT4195: Visual Computing Fundamentals

Digital Image Processing - Assignment 2

September 5, 2016

Aleksander Rognhaugen
Department of Computer and Information Science
Norwegian University of Science and Technology (NTNU)

- **Delivery deadline: September 16, 2016** by 22:00.
- **This assignment counts towards 3 % of your final grade.**
- You can work on your own or in groups of two people.
- Deliver your solution on *itslearning* before the deadline.
- Please upload your report as a PDF file, and package your code into an archive (e.g. zip, rar, tar).
- The programming tasks may be completed in the programming language of your choice, however, it might be a good idea to select one that supports matrix and image processing operations, e.g. MATLAB or Python with NumPy. *The lab computers at IT-S 015 support Python and MATLAB.*
- Your code is part of your delivery, so please make sure that your code is well-documented and as readable as possible.
- For each programming task you need to give a brief explanation of what you did, answer any questions in the task text, and show any results, e.g. images, in the report.
- In this assignment, you can use an existing implementation of the fast Fourier transform (FFT), such as `fft2()` from MATLAB and `numpy.fft.fft2()` from NumPy (Python).

Learning Objectives: Gain a deeper understanding of the convolution operation using the convolution theorem.

1 Theory [0.5 points]

1. **[0.1 points]** The convolution theorem can be seen in Equation 1, where $\mathcal{F}$ is the Fourier transform, $*$ is the convolution operator, and $\cdot$ is pointwise multiplication. What does it entail? Give a stepwise description of how you would perform convolution using the convolution theorem.

$$\mathcal{F}\{f * g\} = \mathcal{F}\{f\} \cdot \mathcal{F}\{g\} \tag{1}$$

2. **[0.1 points]** Convolution kernels are typically understood in terms of the frequency domain. What are high- and low-pass filters?

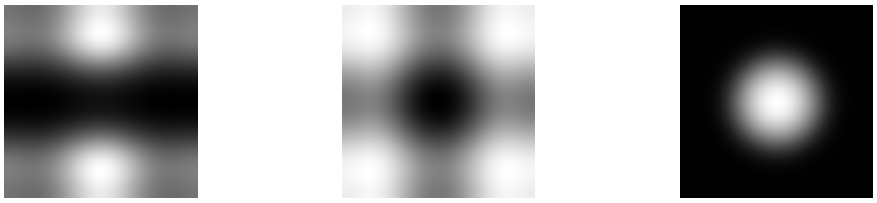
3. **[0.1 points]** The amplitude $|\mathcal{F}\{f\}|$ of three commonly used convolution kernels can be seen in Figure 1. For each kernel (a, b, and c), figure out what kind of kernel it is (high- or low-pass). Explain your reasoning.

4. **[0.1 points]** How will a typical image look like if you remove:

   (i) all the *low* frequencies?

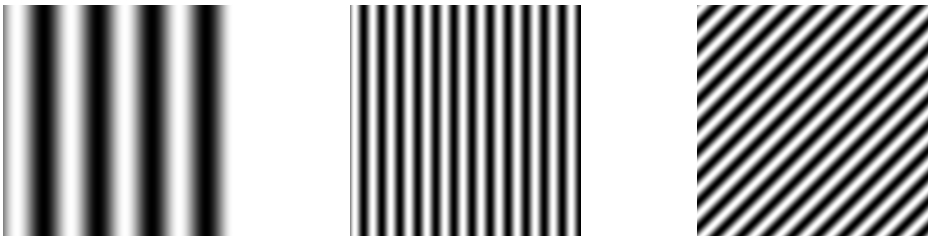
   (ii) all the *high* frequencies?

5. **[0.1 points]** Say we have an image like the one in Figure 2 (a). What would the amplitude look like after the image has been transformed to the frequency domain by the Fourier transform? If we let the frequency of the sinusoidal grating increase as in Figure 2 (b), how does the amplitude change? What if the sinusoidal grating is rotated in the XY plane, as seen in Figure 2 (c)?



(a) (b) (c)

Figure 1: The amplitude $|\mathcal{F}\{f\}|$ of three convolution kernels that have been transformed by the Fourier transform. The zero-frequency, or DC, component have been shifted to the centre for all images. This means that low frequencies can be found around the centre of each image, while high frequencies can be found far from the centre of each image.



(a) (b) (c)

Figure 2: Three images of a sinusoidal grating. (a) and (b) have different frequency, while (c) is rotated in the XY plane.

2 Programming [2 points]

Task 1: Frequency Domain Filtering [1 point]

The Fourier transform is an important signal processing tool that allows us to decompose a signal into its sine and cosine components[1]. A discrete version that samples from the continuous Fourier transform is used on digital images. It does not contain all frequencies, but the number of frequencies sampled are enough to represent the complete image. A 2D version of the discrete Fourier transform (DFT) can be seen in Equation 2. It transforms an $N \times M$ image in the spatial domain to the frequency, or Fourier, domain. The number of frequencies in the Fourier domain is equal to the number of pixels in the spatial domain.

The computational complexity of the DFT is $\mathcal{O}(N^3)$, assuming $N = M$. Thus, the DFT is barely runnable on all but the smallest of images. Fortunately, in 1965, Cooley and Tukey published the first of a slew of algorithms commonly referred to as the fast Fourier transform (FFT)[1].[2] The FFT algorithm reduces the computational complexity of the DFT to $\mathcal{O}(N^2 log_2 N)$, assuming $N = M$.

$$F(u,v) = \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} f(x,y) e^{-i2\pi(\frac{xu}{N} + \frac{yv}{M})} \qquad \text{where} \qquad f(x,y) \in \mathbb{R}^N \times \mathbb{R}^M \tag{2}$$

a) **[0.4 points]** Using an already existing FFT implementation, implement a function that performs convolution using the convolution theorem on greyscale images. Try it out on an appropriate image with both a high- and low-pass convolution kernel. Which convolution kernel to use is up to you. Show the filtered image(s) and the before/after amplitude $|\mathcal{F}\{f\}|$ of the transform in your report. Make sure to *shift* the zero-frequency, or DC, component to the centre before displaying the amplitude.

Sharpening is a technique that strengthens fine details in an image. In essence, this means that an image can be sharpened by adding a high-pass filtered version of the original image to it. This operation can be seen on the left-hand side of Equation 3. The Laplacian (∇^2) is a second order differential operator that is sensitive to fine details in an image, making it ideal for sharpening. A simple approximation to the Laplacian, that does not account for diagonal elements, can be seen on the right-hand side of Equation 3.

$$I_{\text{sharp}} = I_{\text{original}} + (I_{\text{original}} * H_{\text{high-pass}}) \qquad H_{\nabla^2} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \tag{3}$$

b) **[0.2 points]** Implement a function that takes a greyscale image and sharpens it using the Laplacian operator in Equation 3. Perform the convolution using the convolution

[1] Remember that a complex exponent e^{it} can be rewritten in terms of an imaginary sine part and a real cosine part: $e^{it} = \cos t + i \sin t$ (Euler's formula), where $i^2 = -1$.

[2] A similar method was found by Carl Friedrich Gauss around 1800, however, it was left unpublished.

theorem as seen in Equation 1. Apply the function on a greyscale image and show the result in your report.

A hybrid image is an image which has multiple interpretations depending on the viewing distance[2]. The technique was created by Oliva et al. and generates new images by superimposing two images at different spatial scales. In other words, a hybrid image I_{hybrid} is obtained by combining two images I_1 and I_2, where one is filtered with a low-pass filter and the other is filtered with a high-pass filter. This can be seen in Equation 4. The operations are done in the frequency domain.

$$I_{\text{hybrid}} = (I_1 * H_{\text{low-pass}}) + (I_2 * (1.0 - H_{\text{low-pass}})) \tag{4}$$

c) **[0.4 points]** Implement a function that takes two greyscale images and produces a hybrid image. You will need to decide on a point where high and low frequencies are cut off. Show the result using two appropriate images in your report.

Task 2: Aliasing [1 point]

a) **[0.5 points]** Implement a function that downsamples an image by keeping every second row and column from the image. Apply the downsampling function on the TIFF image named `bricks.tiff` accompanying this assignment. Show what happens to the image in your report. What effect are you seeing?

b) **[0.5 points]** Implement a function that takes an image and smooths it using a Gaussian kernel with various kernel sizes, e.g. 3×3, 5×5, and 7×7. Show what happens when the same TIFF image is smoothed before using the downsampling function from (a) in your report.

3 Exam Style Q&A [0.5 points]

Create *one* "exam" style question and answer with the following theme: *Frequency domain.* It should be possible to answer the question in 5-7 minutes. The answer *must* contain all the steps required to get to a solution, and not just the answer itself. Challenging questions are more likely to receive the full amount of points.

You will be asked to add your question and answer – corrected based on any feedback – to a separate document later in the semester.

References

[1] James W Cooley and John W Tukey "An algorithm for the machine calculation of complex Fourier series" in: *Mathematics of computation* 19.90 (1965), pp. 297–301 (cit. on p. 3)

[2] Aude Oliva, Antonio Torralba, and Philippe G. Schyns "Hybrid Images" in: *ACM SIGGRAPH 2006 Papers* SIGGRAPH '06 ACM, 2006, pp. 527–532 DOI: 10.1145/1179352.1141919 (cit. on p. 4)