Image and signal processing: Computer exercise 4
CE04G02T05

PASHEVICH Alexander, SID-LAKHDAR Riyane

15/11/2015

Abstract

This report summarize, explains and refers to the answers we have designed for the fourth "Image and Signal Processing" computer exercise. This paper refers to exploring of spatial frequency scale and spatial structures as well as filters in the Fourier domain. During the work we made experiments, tested the theory and explored the result which is represented below.

1 Introduction

The aim of this report is to describe the relation between the spatial frequency scale of a signal and its spatial structure. This link will be studied thanks to one of the most important tool in signal processing: the Fourier transform. This function is an obvious choice to solve such a problem as it transforms a spacial domain function into a frequency domain function. In this report, we will only consider 2 dimensional discrete functions.

Once we will have considered ways to study the frequency(s) of a signal, we will study a mathematical tool called filters used to transform a signal (sum of signals with different frequencies) into the same signal where some specific constituent signals have been removed (regarding to them frequencies).

2 Material & Methods

2.1 Spatial Frequency Scale and Spatial Structures

2.1.1 Two dimensional Fourier transform: frequency function

The two dimensional Fourier transform of a discrete two dimensional function f is defined as: $\sum_{x=0}^{w} \sum_{y=0}^{h} f(x,y)e^{-i2\pi(ux+vy)}$.
This transform maps a function defined on a spatial domain to a function defined on a frequency domain. In this report, we will try to deduce frequency properties on the 2 dimensional Fourier transform. Then we will try to link this properties with spatial properties on the initial function.

2.1.2 Spatial property

Let S the sum of all the pixels of an image, and let (u_0, v_0) the couple of frequencies such as $F(u_0, v_0) = S$

$$F(u_0, v_0) = \sum_{x=0}^{w} \sum_{y=0}^{h} f(x, y)$$

$$\sum_{x=0}^{w} \sum_{y=0}^{h} f(x, y) e^{-i2\pi(u_0 x + v_0 y)} = \sum_{x=0}^{w} \sum_{y=0}^{h} f(x, y)$$

As the function f and the exponential function are both positive, this equation means that $e^{-i2\pi(u_0 x + v_0 y)} = 1 \ \forall (x, y) \in \mathbb{R}^2$. Thus, the 2 frequency indexes (u,v) where the Fourier Transform of an image equals the sum of all the pixels in the spatial domain are (0, 0).

As we are dealing with finite discrete signal, we can experimentally check this previous property by running the loop Figure Fig 1 on the all the possible input of a Fourier transform and checking if the output of the function corresponds to the expected sum.

```
# return the list of solutions (u, v) such as ft2D(u, v) = s
def solve_discret_equation(ft2d, s):
    res = []
    for x in xrange(len(ft2d)):
        for y in xrange(len(ft2d[0])):
            if ft2d[x, y] == s:
                res.append((x, y))
    return res
```

Figure 1: Python code to solve an equation on a 2D function.

2.1.3 Frequency property

The Fourier transform of an image may be interpreted using several complex number representation. The one we will mainly use in this report is the exponential one: (magnitude, phase).
Let consider the spectrum operator. This mathematical operator computes the logarithm of the squared modulus (magnitude) of complex function. The program in the figure 2 computes the spectrum of a given 2D complex input function (2D Fourier transform).

In the "Experiments" section, we will see how this algorithm, applied to different images (with different spatial properties) allows us to determine the frequency properties of this pictures and to decompose an image into signals depending on them frequencies.

2.2 Filters in the frequential domain

2.2.1 Filters' descriptions

The filters used in the computer exercise are defined below.
The first filter is called cylinder and it is defined with the formula 1. It has one

```
def power_spectrum_2D(ft2d, fx=None, fy=None):
    epsilon = np.finfo(float).eps
    if fx != None:
        res = np.zeros(len(ft2d[0]))
        for y in xrange(len(ft2d[0])):
            val             = ft2d[fx, y]
            magnitudeSquare = val.imag * val.imag + val.real * val.real
            res[y]          = np.log(magnitudeSquare + epsilon)
        return res

    if fy != None:
        res = np.zeros(len(ft2d))
        for x in xrange(len(ft2d)):
            val             = ft2d[x, fy]
            magnitudeSquare = val.imag * val.imag + val.real * val.real
            res[x]          = np.log(magnitudeSquare + epsilon)
        return res

    res = np.zeros(ft2d.shape)
    for x in xrange(len(ft2d)):
        for y in xrange(len(ft2d[0])):
            val             = ft2d[x, y]
            magnitudeSquare = val.imag * val.imag + val.real * val.real
            res[x, y] = np.log(magnitudeSquare + epsilon)
    return res
```

Figure 2: Python code to compute the spectrum of a 2D function.

parameter which is called cut off frequency (f_c).

$$H_F(f_x, f_y) = \begin{cases} 1 & \text{if } \sqrt{f_x^+ f_y^2} < f_c \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

The Butterworth filter is defined with formula 2. In addition to the cut off frequency it also has a parameter of order (n).

$$H_F(f_x, f_y) = \frac{1}{1 + (\frac{\sqrt{x^2 + f_y^2}}{fc})^{2n}} \tag{2}$$

The last filter is called gaussian and it is one of the most known and widely used filters. It has two parameters: sigmas for x and for y which are standard deviations for both axes.

$$H_F(f_x, f_y) = e^{-\frac{f_x^2}{2\sigma_x^2} - \frac{f_y^2}{2\sigma_y^2}} \tag{3}$$

The code which implements those filters is given in figure 3. To apply those filters to original image I we use the formula:

$$I_{filtered} = F^{-1}(H) \text{ where } H(f_x, f_y) = H_F(f_x, f_y) * F(I)(f_x, f_y) \tag{4}$$

If we would like to introduce the notion of cut off frequency while working with gaussian filter, we would like the following formula to be true $H_F(f_c, 0) = H_F(0, f_c) = \frac{1}{2}$. From the formula we can deduce the value of $\sigma_x = \sigma_y$.

$$H_f(f_c, 0) = e^{-\frac{f_c^2}{2\sigma_x^2}} = \frac{1}{2} = e^{ln\frac{1}{2}} \Rightarrow \sigma_x = \frac{f_c}{\sqrt{2ln2}}$$

Exactly the same applies to σ_y so in this case $\sigma_x = \sigma_y$.

2.2.2 Implementation

```
def cylinder_filter(Nx, Ny, cutOffFrequency, typeFilter):
    res = np.zeros((Nx, Ny))
    for x in range(Nx):
        for y in range(Ny):
            fx = 1. * x / Nx
            fy = 1. * y / Ny
            if np.sqrt(fx ** 2 + fy ** 2) <= cutOffFrequency:
                res[x, y] = 1
            else:
                res[x, y] = 0
    return res


def butterworth_filter(Nx, Ny, cutOffFrequency, n):
    res = np.zeros((Nx, Ny))
    for x in range(Nx):
        for y in range(Ny):
            fx = 1. * x / Nx
            fy = 1. * y / Ny
            res[x, y] = 1. / (1 + (np.sqrt(fx ** 2 + fy ** 2) /
                cutOffFrequency) ** (2 * n))
    return res


def gaussian_filter(Nx, Ny, sigma_x, sigma_y):
    res = np.zeros((Nx, Ny))
    for x in range(Nx):
        for y in range(Ny):
            fx = 1. * x / Nx
            fy = 1. * y / Ny
            res[x, y] = np.exp(- 0.5 * (fx ** 2) / (sigma_x ** 2) - 0.5 *
                (fy ** 2) / (sigma_y ** 2))
    return res
```

Figure 3: Python code for filters' implementation

3 Experiments

3.1 Experimental setup

In this report, we only use the provided pictures. This images and some of them graphical properties will be given in this section. We will also use our previously defined algorithm to find some of the frequency properties.

3.2 Spatial Frequency Scale and Spatial Structures

3.2.1 Fourier transform: symmetry on image

For images, generally, the 2D Fourier Transform is centered around zero. This symmetry property is clearly shown on the figure Fig **??** (right picture).
In fact this symmetry is not according to the central coordinate of the picture (transcribed to frequencies) but is according to a period of size (w, h) where w is the width of the picture and h its height. However, this picture clearly shows that a period starts at (0, 0). Thus, in the following spectrum we will describe, we will consider power spectrum in the usual form (the null frequency at the center of the Fourier matrix).

3.2.2 Spectrum interpretation

First of all, we applied the spectrum program given in the "Material & methods" section to the picture in the figure 4 (left picture). The result is represented in the same figure (right picture).

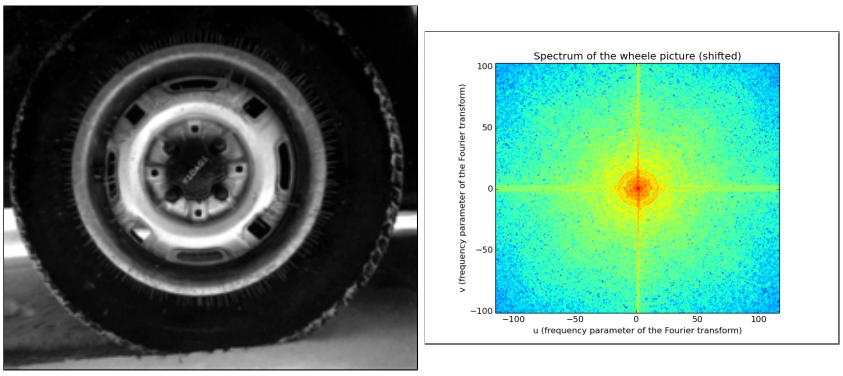

Figure 4: Black and white picture and its corresponding spectrum

An image is the sum of signals s_ϕ where ϕ is the frequency of the signal. We can observe on this spectrum that the small frequencies (u or v close to zero) have a very important magnitude compared to the others. This means that small frequency signals that compose the initial image are much more significant that the high frequency signals. What we mean here by "significant" is that mainly the original image consists of those low frequencies (of the 2D signals which correspond to them) and if we remove the high frequencies we will not loose a lot of information. So removing the low frequencies signals from the image means lost of a lot of details while removing other signals (with a higher frequency) would have less impact on the quality of the image.

After we retrieved the Fourier coefficients and calculated the magnitude, we applied the inverse Fourier transform and received the same image which is not surprising actually as according to the theory $I = F^{-1}(F(I))$.

Let's now consider the picture and its spectrum in the figure 5. This picture has some obvious spatial periodicity properties. But how does this properties affect the frequency properties of the picture?

First of all, we can notice that the spectrum alternates high and low magnitudes over f_x (horizontal frequency axes) and f_y (vertical frequency axes) with the same fixed period. This allows us to determine frequency coordinates for the first frequency peak at $(f_x = 15, f_y = 6)$.

Secondly, we can notice that the red color is highly dominant (frequencies with high magnitude). Thus the contrast between colors is extremely high in most of the picture. We also notice extremely low magnitude (periodically) which correspond to spaces where the picture has the same color with no contrast. Finally, we can notice that the alternation between high and low magnitudes (over horizontal or vertical axes) are not strict (there are shadings between red and blue magnitudes). Thus, the break lines between white and black colors in

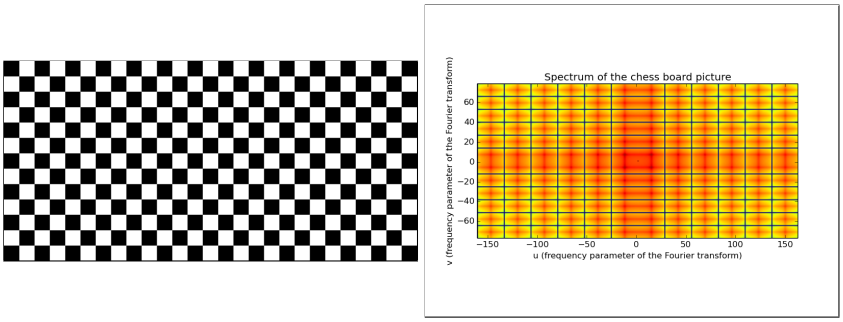

Figure 5: Black and white picture and its corresponding spectrum

the original picture are not strict and contain gray pixels.

3.2.3 Fixing a frequency in the spectrum: 1D frequency function

For a given fixed frequency (f_x or f_y), we want now to determine the behavior of the Fourier transform depending on the other frequency. Using the program described in the "Materials & Methods" section, we have represented on the figure 6 the spectrum power when we have fixed the f_x frequency (left figure) and the f_y frequency (right figure).

This figures confirm the results of the 2D spectrum (figure 5): For the line

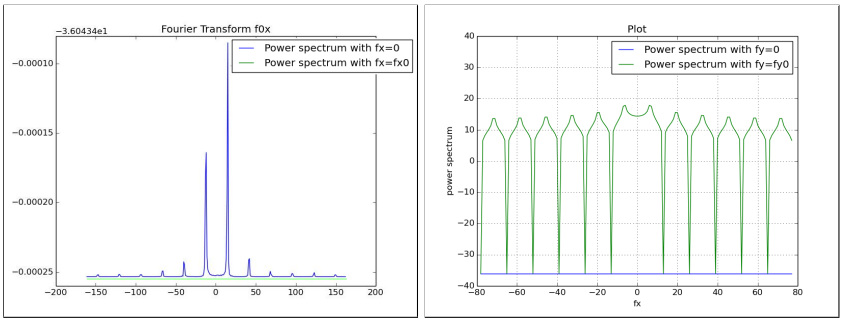

Figure 6: 1D spectrum with a fixed second parameter (chess board image)

($f_y = 0$) of the spectrum, the peak and the holes of the spectrum are perfectly matching the peak and the holes of the 1D function for $f_y = 0$ fixed. This checking may successfully be done for all the other value of fixed f_x and f_y.

3.3 Filters in the frequential domain

When we apply filters to an image, we changes Fourier coefficients using the formula 4. Normally, we remove either high frequencies (*Low pass filters*) or low frequencies (*High pass filters*) by removing (setting them to null) Fourier coefficients with high or low indexes correspondingly. Some filters don't completely remove those coefficients but instead of it they multiply the coefficients by very small number. If we remove high indexes, we see only low frequencies

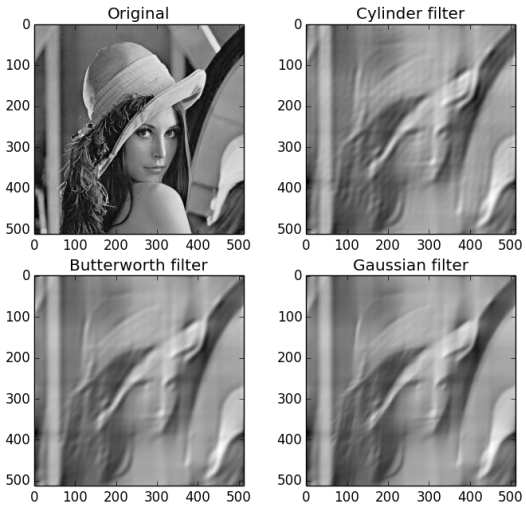

Figure 7: Original and filtered images

which means that the image looses clearness and becomes blurred. However we can still see the main contours. In case of high pass filters, we do exactly the opposite. We loose most of the sensible information however acquire some additional information about rapid drops of color for example. Anyway in both cases we loose some information on the image in order to concentrate on different aspects of it.

We can see the results of applying of the cylinder filter which is given by formula 1 on the classical image of Lena on the figure 7. We used the filter with cut off frequency $f_c = \frac{1}{16}$. As we can see, the image became blurred however we still can understand what is shown on it. We also show the image filtered by gaussian and butterworth filters and also the original one. As we can see, the gaussian and butterworth filters do pretty much the same that the cylinder. They blur the image by reducing high frequencies. However, they do not completely remove them. Instead of this they reduce values of Fourier coefficients for high frequencies which actually do pretty much the same. Filters are represented in the figure 8. Those images are plots for H_F with fixed $f_y = 0$. From the image it is clear that the cylinder filter just removes every frequency which is higher than some f^x_{max}. If we look at the formula 1, we can deduce that $f^x_{max} = f_c * N_x = \frac{512}{16} = 32$ for our case. The rest of frequencies stay without changes. In case of butterworth and gaussian filters the filter becomes smooth (it means that we can take derivative of it which is big advantage). However, as we can deduce from the figure the butterworth filter does almost the same with the cylinder. Frequencies higher than 30-40 are almost neglected (which also corresponds to $f_c * N_x$). Gaussian filter takes into account frequencies lower than 90.

1D impulse responses are represented in the figure 9. The pictures are the

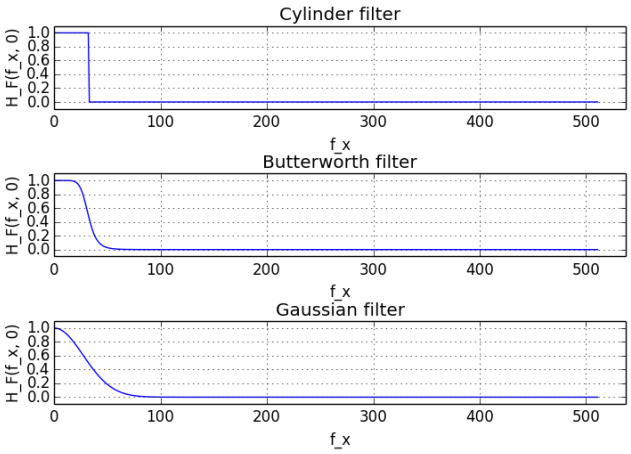

Figure 8: Filters

same but on the second only first 50 samples are shown for the sake of zooming in. As it can be seen, in the original image most of frequencies are negligibly small. However, the highest frequencies are quite dense and the filters completely diminish them which makes the filtered image low blurred. We also can deduce from the image that the filters do pretty much the same job without looking at the outputs.

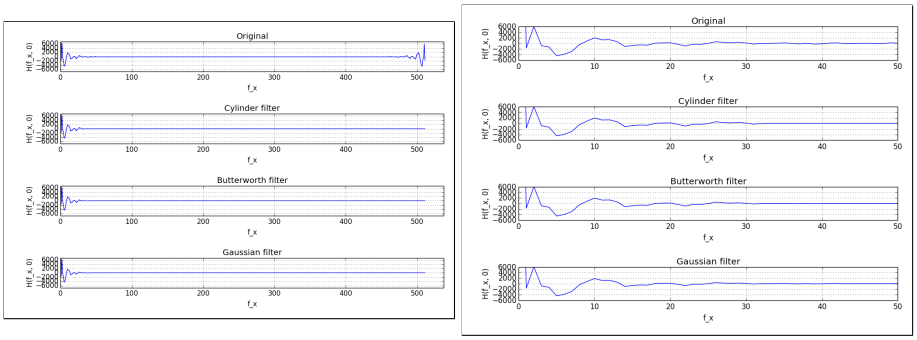

Figure 9: 1D impulse responses for the filters

4 Conclusion

During the work on the lab exercises, we worked with Fourier transform and explored spatial properties of some images. We also used filters to change the spatial properties of input images. We implemented the used methods using Python programming language as well as understood how Fourier transform of 2D signals work and what different frequencies and their magnitude means. To conclude, this lab exercises gave us insight how Fourier transform of 2D signals works and what their spatial properties mean.