Digital Image Processing
- Task 2-
—
Filter Design

Multimedia Understanding Group
Department of Electrical and Computer Engineering
Aristotle University of Thessaloniki

Spring 2018

Quotation marks

In the second assignment of the course you will implement the following:

1. Filtering process in the frequency domain
2. Design of lowpass, highpass and bandpass filters
3. Design of directional filters

Along with the pronunciation you will also find the MATLAB help filedip_hw_2.mat which includes the data me that you will use in each query such as input images and filters.

Familiarity with spatial frequencies

Consider the DFT of an image $f(m_1, m_2)$ with dimensions $N \times N$ as follows:

$$F(you_1, u_2) = \begin{cases} -1 + 0j & , if \quad you_1 = m \ and \quad you_2 = l \\ 1 + 0j & , if \quad you_1 = N - m \quad and \ you_2 = N - l \\ 0 + 0j & , else \end{cases} \tag{1}$$

for some integers m and l as well as $0 \leq you_1, u_2 \leq N - 1$. Essentially this corresponds to a table of DFT which has values everywhere equal to 0 except for two **symmetrically** placed coefficients. You can see 2 para-samples in Figure 1. Equation 1 is valid for DFT matrices of even and odd size.

If we calculate the inverse DFT $f(m_1, m_2)$ of $F(you_1, u_2)$ then the periodic tables appear in the field of space cosines in the form of cosines with the propagation direction in the direction of the vector $[m, l]$ (or $[N - m, N - l]$). The frequency of the cosine depends on the length of the vector mentioned earlier. Figure 2 shows the above procedure applied to the DFT plots of the example in Figure 1.

You are invited to experiment and become familiar with the above. Specifically, perform the following experiment:

First, construct a DFT table $X(you_1, u_2)$ with a pair of non-zero symmetric coefficients as well as non-zero DC component by making appropriate use of Equation 1. Then apply the routine ifft2 her MATLAB to get the original image $x(m_1, m_2)$ in the field of space. Then, multiply the $x(m_1, m_2)$ with the function $(-1)^{m_1+m_2}$ to transfer the (0,0) of DFT in the center of the table. Finally, apply

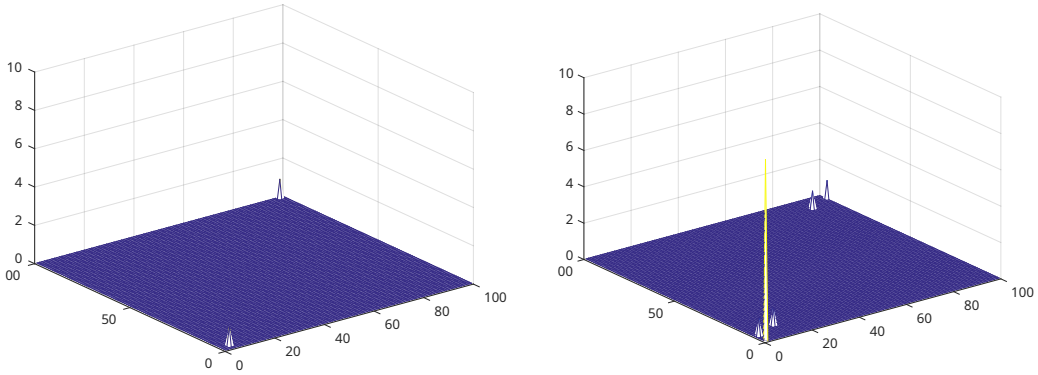

Figure 1: Examples of two DFTs. A) a pair of symmetric coefficients without DC component (left), B) two pairs of symmetric coefficients plus DC component (right).

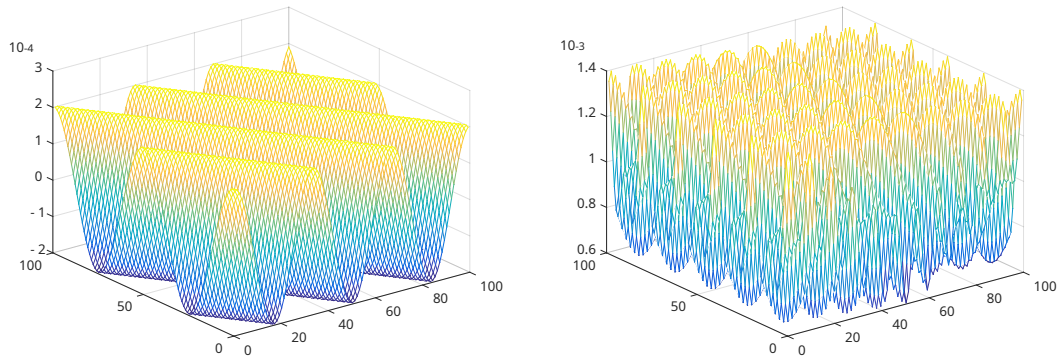

Figure 2: The inverse DFTs of examples A) and B) of Figure 1, left and right respectively. Note the effect of the extra pair of symmetric coefficients (presence of an extra cosine) as well as the DC component (increased amplitude).

the routinefft2 on the centered table to get the transformationFourier. What do you observe? To what positions have the non-zero coefficients shifted? To what position has the DC component shifted? **Note/ warning:**In contrast to the theory where array indices start from (0,0), in MATLAB numbering starts from (1,1).

1 Filtering in the frequency domain

Construct the function:

```
1   functionimOut = myFiltFreq(imIn , someFilt)
```

where
imIn: The input image
someFilt: The description of the filter in the frequency domain imOut:
The output image with the result of the filtering process

The functionmyFiltFreq implements the process of filtering an input image with a filter in the frequency domain. frequency. As part of your assignment, you are asked to implement the algorithm described in the course lectures, specifically in lecture7, page33[1].

Note/warning: In the context of the work, the filters as well as the input images must have the same dimensions and be square. For example, if the input image isimIn has dimensions$M \times M$, then the filter someFilt must have dimensions$M \times M$.

[1]http://alexander.ee.auth.gr:8083/eTHMMY/archive/118/downloadFile/5991/DIP%20notes%20Lecture-7%202015.pdf

2 Filter Design

2.1 Design of Low-Pass Filters

Construct the following functions that design and return low-pass filters of the type: a)*Ideal Deep sea*b) *Butterworth*and c)*Gauss*.

```
1  functionfiltOut = myLowPassIdeal(cutOff , M)
```

where
cutOff: The cutoff frequency (cutoff frequency)
M: The dimension of the (square) filter
filtOut: The description of the ideal low-pass filter in the frequency domain

```
1  functionfiltOut = myLowPassButterworth(cutOff , n, M)
```

where
cutOff: The cutoff frequency n:
The order of the filter
M: The dimension of the filter
filtOut: The description ofButterworth low-pass filter in the frequency domain

```
1  functionfiltOut = myLowPassGauss(sigma , M)
```

where
sigma: The dispersion of the function that determines the cutoff frequency M: The
dimension of the filter
filtOut: The description of the deep passGaussian filter in the frequency domain

2.2 High-pass and Band-pass Filters

Next, construct the function that produces the corresponding high-pass filter, more specifically:

```
1  functionfiltOut = myHighPassIdeal(cutOff , M)
```

where
cutOff: The cutoff frequency M:
The dimension of the filter
filtOut: The description of the ideal high-pass filter in the frequency domain

```
1  functionfiltOut = myHighPassButterworth(cutOff , n, M)
```

where
cutOff: The cutoff frequency n:
The order of the filter
M: The dimension of the filter
filtOut: The description ofButterworth high-pass filter in the frequency domain

```
1  functionfiltOut = myHighPassGauss(sigma , M)
```

where
sigma: The dispersion of the function that determines the cutoff frequency M: The
dimension of the filter
filtOut: The description of the high-passGaussian filter in the frequency domain

Finally, construct the routines that construct the corresponding bandpass filters, more specifically:

```
function filtOut = myBandPassIdeal(cutOffLow , cutOffHigh , M)
```

where
cutOffLow: The low cutoff frequency
cutOffHigh: The high cutoff frequency M:
The dimension of the filter
filtOut: The description of the ideal bandpass filter in the frequency domain

```
function filtOut = myBandPassButterworth(cutOffLow , cutOffHigh , n, M)
```

where
cutOffLow: The low cutoff frequency
cutOffHigh: The high cutoff frequency n:
The order of the filter
M: The dimension of the filter
filtOut: The description ofButterworth bandpass filter in the frequency domain

```
function filtOut = myBandPassGauss(sigmaLow ,sigmaHigh , M )
```

where
sigmaLow: The dispersion of the function that determines the low cutoff frequency
sigmaHigh: The dispersion of the function that determines the high cutoff frequency M: The
dimension of the filter
filtOut: The description of the lane crossingGaussian filter in the frequency domain

3 Directional Filters

Construct the following routines that design and return directional filters of the following types: a) Ideal, b) Butterworth and c) Gauss and of the following categories: a) lowpass, b) highpass and c) bandpass.

```
function filtOut = myLowPassIdealDir(cutOff , M, theta , phi)
```

where
cutOff: The cutoff frequency (cutoff frequency)
M: The dimension of the (square) filter theta: The
orientation of the filter, in degrees
phi: The size of the circular sector (essentially the “opening” of the filter), in degrees filtOut: The
description of the ideal low-pass directional filter in the frequency domain

```
function filtOut = myLowPassButterworthDir(cutOff , n, M, theta , phi)
```

where
cutOff: The cutoff frequency n:
The order of the filter
M: The dimension of the filter
theta: The orientation of the filter, in degrees
phi: The size of the circular sector (essentially the “opening” of the filter), in degrees
filtOut: The description ofButterworth low-pass directional filter in the frequency domain

```
function filtOut = myLowPassGaussDir(sigma , M, theta , phi)
```

where
sigma: The dispersion of the function that determines the cutoff frequency M: The
dimension of the filter
theta: The orientation of the filter, in degrees
phi: The size of the circular sector (essentially the “opening” of the filter), in degrees
filtOut: The description of the lowpassGaussian filter in the frequency domain

```
1  functionfiltOut = myHighPassIdealDir(cutOff , M, theta , phi)
```

where
cutOff: The cutoff frequency M:
The dimension of the filter
theta: The orientation of the filter, in degrees
phi: The size of the circular sector (essentially the “opening” of the filter), in degrees filtOut:
The description of the ideal high-pass directional filter in the frequency domain

```
1  functionfiltOut = myHighPassButterworthDir(cutOff , n, M, theta , phi)
```

where
cutOff: The cutoff frequency n:
The order of the filter
M: The dimension of the filter
theta: The orientation of the filter, in degrees
phi: The size of the circular sector (essentially the “opening” of the filter), in degrees filtOut: The
description ofButterworth high-pass directional filter in the frequency domain

```
1  functionfiltOut = myHighPassGaussDir(sigma , M, theta , phi)
```

where
sigma: The dispersion of the function that determines the cutoff frequency M: The
dimension of the filter
theta: The orientation of the filter, in degrees
phi: The size of the circular sector (essentially the “opening” of the filter), in degrees filtOut: The
description of the highpassGaussian directional filter in the frequency domain

```
1  functionfiltOut = myBandPassIdealDir(cutOffLow ,cutOffHigh , M, theta , phi)
```

where
cutOffLow: The low cutoff frequency
cutOffHigh: The high cutoff frequency M:
The dimension of the filter
theta: The orientation of the filter, in degrees
phi: The size of the circular sector (essentially the “opening” of the filter), in degrees filtOut: The
description of the ideal bandpass directional filter in the frequency domain

```
1  functionfiltOut = myBandPassButterworthDir(cutOffLow , cutOffHigh , n, M,
       theta, phi)
```

where
cutOffLow: The low cutoff frequency
cutOffHigh: The high cutoff frequency n:
The order of the filter
M: The dimension of the filter

theta: The orientation of the filter, in degrees
phi: The size of the circular sector (essentially the “opening” of the filter), in degrees
filtOut: The description ofButterworth bandpass directional filter in the frequency domain

```
function filtOut = myBandPassGaussDir(sigmaLow ,sigmaHigh , M, theta , phi)
```

where
sigmaLow: The dispersion of the function that determines the low cutoff frequency
sigmaHigh: The dispersion of the function that determines the high cutoff frequency M: The dimension of the filter
theta: The orientation of the filter, in degrees
phi: The size of the circular sector (essentially the “opening” of the filter), in degrees filtOut: The description of the bandpassGaussian directional filter in the frequency domain

The geometric interpretation of the parameters i and ϕ can be seen in Figure 3

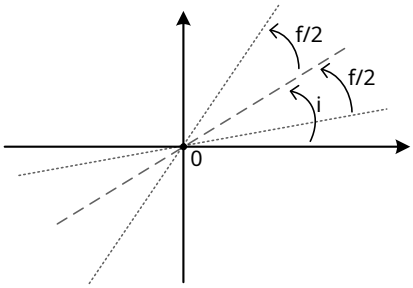

Figure 3: Geometric interpretation of angles i and ϕ.

Evaluation & deliverables

When submitting the assignment, you must deliver the files with the functions:

- myFiltFreq.m
- myLowPassIdeal.m
- myLowPassButterworth.m
- myLowPassGauss.m
- myHighPassIdeal.m
- myHighPassButterworth.m
- myHighPassGauss.m
- myBandPassIdeal.m
- myBandPassButterworth.m
- myBandPassGauss.m
- myLowPassIdealDir.m
- myLowPassButterworthDir.m
- myLowPassGaussDir.m
- myHighPassIdealDir.m

Table 1: Parameter table for designing the module filters2

Type	Class	cutOff	cutOffLow	cutOffHigh	M	n	sigma	sigmaLow	sigmaHigh
Ideal	Lowpass	50	-	-	500	-	-	-	-
Butterworth	Lowpass	50	-	-	500	5	-	-	-
Gauss	Lowpass	-	-	-	500	-	50	-	-
Ideal	Highpass	150	-	-	500	-	-	-	-
Butterworth	Highpass	150	-	-	500	5	-	-	-
Gauss	Highpass	-	-	-	500	-	150	-	-
Ideal	Bandpass	-	50	150	500	-	-	-	-
Butterworth	Bandpass	-	50	150	500	5	-	-	-
Gauss	Bandpass	-	-	-	500	-	-	50	150

- myHighPassButterworthDir.m
- myHighPassGaussDir.m
- myBandPassIdealDir.m
- myBandPassButterworthDir.m
- myBandPassGaussDir.m

as well as a report.**Additional**, per section you must also submit a script nameddemo1.m, demo2.m, and demo3.m respectively, which will be executed**without**arguments and will present the requirements of each unit (see the following paragraphs). In the report you should also present any design choices you have made.

Section 1

For the module demo1 you are invited to present the operation ofmyFiltFreq using the environment writing a filter that will be given to you within theMATLAB file with the auxiliary material. Specifically, load it into MATLAByour workspace variables**someFreqFilt**and**demo1Im**which is the description of a filter in the field of frequency and the image to which we will apply the filter. Then call the filtering routine you constructed with the following arguments:imOut = myFiltFreq(demo1Im, someFreqFilt). What observations do you make? Based on the visual effect of the filtered image, what do you think the category of the filter is?

The above functions should be inside the scriptdemo1.m.

Section 2

For the module demo2 you are invited to present the operation of low-pass/high-pass and band-pass filters. of the Ideal/Butterworth and Gauss type filters. Essentially you will construct the whole9 filters. In the Table 1 you will find the parameters of the filters you will build. Then, load them intoMATLAByour workspace variables**demo2Im1**and**demo2Im2**which are essentially the2 images to be processed by our filters. Present the results for each input image and filter combination (from Table 1). It is understood that for the filtering process it should be performed with the routinemyFiltFreq which you made in the section1.

The above functions should be inside the scriptdemo2.m.

Section 3

For the last demo, you are invited to present the operation of directional filters using the routines you constructed according to the section3. Specifically starting with $i = 0^\circ$ and fixed "opening" $\phi = 30^\circ$ construct6 **non-overlapping** ideal low-pass directional filters with dimension M=500 and cutOff =200. The union of these 6 filters should contain every angle from 0° until 360° (tip: remember symmetry).

Then, construct the final ideal low-pass directional filter in the following way:

$$D = \sum_{I=1}^{6} d_I \tag{2}$$

where D is the $M \times M$ description of **final** directional filter and d_I the $M \times M$ descriptions of the individual.

Apply the final directional filter D in the pictures **demo2Im1** and **demo2Im2**. Compare and discuss in the report the result of the above procedure in relation to filtering with the corresponding Ideal low-pass filter. It is understood that in each case the filtering process will be done with the routinemyFiltFreq that you constructed within the framework of the unit1.

Repeat the above process. (i.e. construction of a directional filter and comparison with the corresponding non-directional) for each combination of filter type and category (total9 times). The directional (and non-directional) filters that you will build and compare should have specifications according to Table 1.

The above functions should be inside the scriptdemo3.m.

About submitting the work

Submit a report with the descriptions and conclusions requested in the presentation. The report should demonstrate the correct operation of your code in the images provided.

The code should be commented so that it is clear what exactly it does (at a theoretical level, not at the function call level). The code should also be executable and calculate the correct results for *any* input meets the assumptions of the utterance, and not just for the images you are given.

Necessary conditions for grading your work are that the code executes without error, as well as that the following are met:

- Submit a single file, zip type.
- The file name must be AEM.zip, where AEM is the four digits of the student's AEM. group.
- The file to be submitted must contain the Maltab code files and the filereport.pdf which will is the work report.
- The report must be a PDF file, and have a namereport.pdf.
- All code files must be UTF-8 text files, and have the extension .m.
- The zip file you submit must not contain any folders.
- Do not submit the images given to you for experimentation.
- Do not submit files that are not needed for the operation of your code, or folders/files that your operating system creates, e.g. "Thumbs.db", ".DS_Store", ".directory".
- For the naming of the files contained in the file to be submitted, use only English characters, and not Greek or other symbols, e.g. "#", "$", "%" etc.