M1 E3A
Lab 1

Signal and image processing - M1 E3A - UEVE

Preamble

The Practical Work of the "Signal, image" module will be carried out in Matlab using in particular the *Image Processing* toolbox. The latter's functions can be listed by typing **help images** in the Matlab command editor. To obtain a detailed description of the functions, use the contextual help by typing **help** followed by the name of the function or use the online help by typing **helpdesk**. Before going directly to the topic of the lab, you can discover the demonstrations associated with the *Image Processing toolbox* by typing **demos**.

A given image is defined as a two-dimensional finite signal sampled with quantized value in a given color space. The elements of an image are called "pixels" (*picture element*). So :

- an image has finite dimensions (finite signal);
- an image is characterized by two spatial dimensions (two-dimensional signal);
- the pixels are generally arranged in a rectangular grid (sampled signal);
- finally, the pixel values belong to a discrete space (quantized values).

In matlab:

- A grayscale image is considered as a 2-dimensional array whose elements $I(i, j)$ are ordered from top to bottom and left to right. Indexing of rows and columns starts at 1. If I is the name of the image, then $I(1, 1)$ returns the value of the pixel with coordinates $(1, 1)$. $I(:, 1)$ returns the values of the pixels in the first column, and $I(1, :)$ returns the values of the pixels in the first row.

- Arithmetic operations on arrays can be performed in a matrix fashion with the operators: "*", "/" or item by item with the operators: ". *", "./".

- Matlab supports 4 image formats: binary images, intensity images (grayscale), RGB color images, indexed color images. It is possible to change the format using the following functions: **ind2gray**: indexed to intensity, **ind2rgb**: indexed to RGB, rgb2ind: RGB to indexed, **rgb2gray**: RGB to intensity, **rgb2bw**: RGB to binary, etc.

- The values of the pixels of the images can be typed: logical (0 or 1 for binary images), unsigned integer coded on 8 bits (between 0 and 255), unsigned integer coded on 16 bits (between 0 and 65535), real (between 0.0 and 1.0). It is possible to change the type of variables using the following functions: **im2double**, **im2uint8**, **im2uint16**, **uint8**, **uint16**.

- The display of an image can be done, either in real and the pixel values are then between 0.0 and 1.0, or in full and the pixel values are then included, by example, between 0 and 255 for unsigned 8-bit integers.

- Loading an image of type uint8 (eight-bit integer) into memory is done with the **imread** function. It is essential to transform it into a double (real floating point coded on 64 bits) using the **double** function.
- The image display is performed by the **imshow** function.
- The **imwrite** and **print** functions allow saving, respectively, images and figures in different formats (tif, jpg, bmp, pcx, png, gif, emf, eps, ...).

2D discrete TF and frequency domain filtering

Exercise 1 : Interpretation of frequency content

1. The **fft2** function allows you to calculate the Fourier transform of an image and the **ifft2** function the inverse transform. The **fftshift** function allows you to center a Fourier transform on the fundamental (zero spatial frequency). Calculate the centered Fourier transform of an image of your choice in grayscale. Display the module or the real part of this transform either in an image or in a graphic (surf function, image, imagesc, imshow, etc.). We will use: $log(1 + abs(fft2(I))$.
2. Where are the low frequencies, the high frequencies, the coefficients having the greatest amplitude? Comment on the spectral content. Test on several images. Comment according to the frequency content of the images.
3. Find the average of the gray levels of the image using the Fourier transform.

Exercise 2 : Properties of the Fourier transform

Check the following properties of the discrete 2D Fourier transform on the image of your choice. When necessary, be sure to create the appropriate images in terms of content but also in terms of size.

1. Linearity. Warning: please handle typing correctly before each arithmetic operation.
2. Rotation. Hint: we can use **imrotate**.
3. What is the result of the Fourier transform of $I(x, y) \times (-1)^{x+y}$? What property of the Fourier transform is revealed?

Exercise 3 : Fourier transform of basic images

Calculate the 2D DFT of the following images:

1. A sinusoid function : $sin(x \times y)$ then $sin(x)$ and $sin(y)$. Note: A sinusoidal image in grayscale is obtained by: $I(x, y) = 255 \times \frac{(sin(2\pi(Ax+By))+1)}{2}$
   A and B are the frequencies (inverse of the period) for the vertical and horizontal displacement of the sine respectively. Test for several values of A and B (0, 1, 128/256, 64/256, 32/256, 16/256, etc.).
2. A gate function (image of a rectangle).
3. A Gaussian (use **fspecial**).
4. A Uniform average filter (use **fspecial**). Compare with Gaussian in terms of frequency content.
5. A Laplacian filter (use **fspecial**). For each case, comment the obtained results. Make a link with the theoretical results seen in class. Do not hesitate to vary the parameters and to refer to the figure of the appendix to implement the *padding* for the generated filters of reduced size.

Exercise 4 : Fourier transform and convolution

Show that a convolution in the spatial domain is equivalent to a multiplication in the frequency domain. Demonstrate this property with a Laplacian (as it was generated in the previous exercise). Comment on the obtained result.

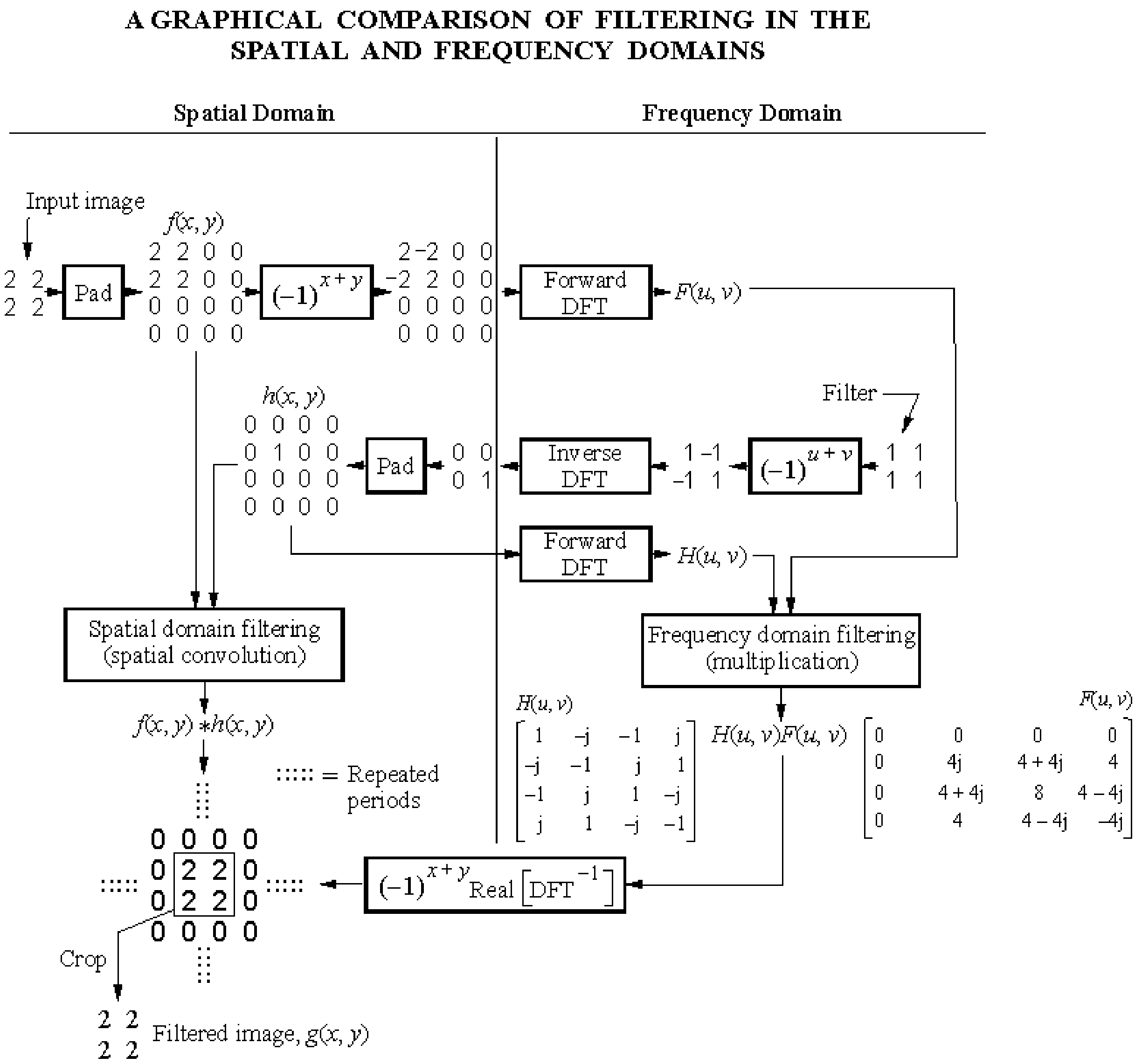
Exercise 5 : Sampling

1. Calculate the Fourier transform of a pulse train of the period of your choice. To generate an image 256×256 with a pulse train of period 4 pixels:
```
A = zeros (4,4);
A (1.1) = 1;
B = repmat (A, 256/4,256/4);
```
2. Sample the image of your choice then the UCSB image using the generated pulse train. Note: multiply the image with the pulse train.
3. Calculate the 2D DFT of the sampled image. Make the link with the theoretical results seen in class.
4. Suggest a way to find the original image from the sampled image knowing that the sampling highlights the phenomena of Aliasing. You are not asked to implement the proposed solution.

Exercise 6 : Filtering in the frequency domain

1. Implement a low-pass filtering then a high-pass filtering on the image of your choice. What do these two operations achieve? We will compare ideal filters with Butterworth filters, taking care to vary the cutoff frequencies (0.25, 0.5, etc.). Reminder: $H(u,v) = \frac{1}{1+(D(u,v)/D_0)^{2n}}$ where n is the order of the filter, D_0 the frequency cutoff and $D(u,v) = \sqrt{u^2+v^2}$.

Annex


A GRAPHICAL COMPARISON OF FILTERING IN THE SPATIAL AND FREQUENCY DOMAINS
Spatial Domain
Frequency Domain
Input image
f(x, y)
Pad
(−1)^{x+y}
Forward DFT
F(u, v)
h(x, y)
Filter
Inverse DFT
(−1)^{u+v}
H(u, v)
Spatial domain filtering (spatial convolution)
Frequency domain filtering (multiplication)
f(x, y) * h(x, y)
H(u, v)F(u, v)
::::: = Repeated periods
(−1)^{x+y} Real [DFT^{−1}]
Crop
Filtered image, g(x, y)