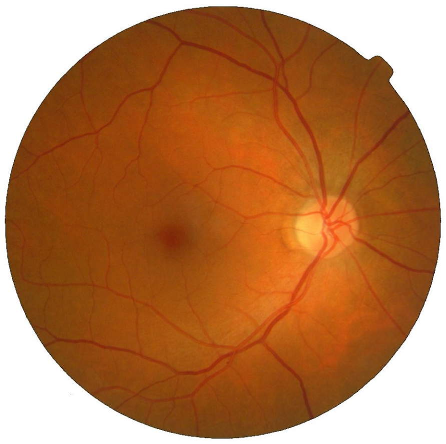
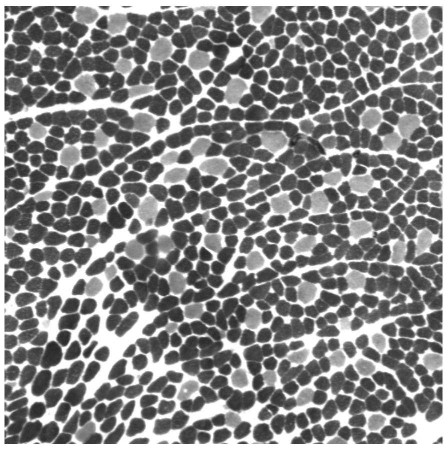
Tutorial: Introduction to image processing

In this tutorial, you will discover the basic functions in order to load, manipulate and display images. The main informations of the images will be retrieved, like size, number of channels, storage class, etc. Afterwards, you will be able to perform your first classic filters.

The different processes will be realized on the following images:


(a) Retinal vessels.


(b) Muscle cells.

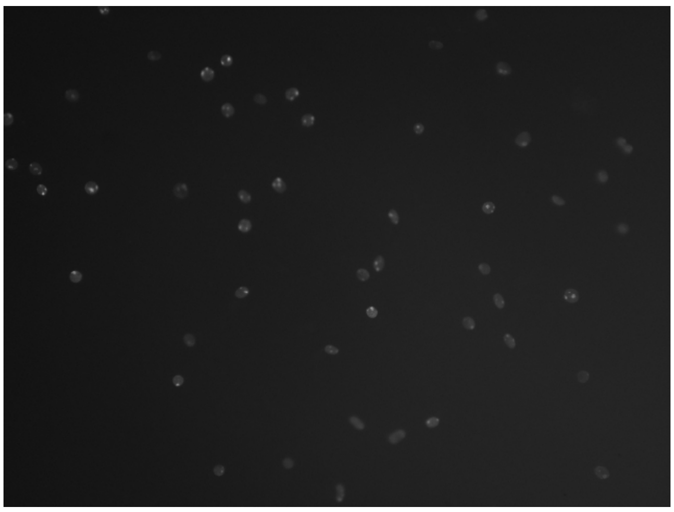

(c) Cornea cells (BIIGC, Univ. Jean Monnet, Saint-Etienne, France).

Figure 1: Image examples.

1 First manipulations

Image loading can be made by the use of the python function imageio.imread. The visualization of the image in the screen is realized by using the module matplotlib.pyplot.

Image loading can be made by the use of the MATLAB® function imread. The visualization of the image in the screen is realized either by the MATLAB® function imagesc or imshow.

- Load and visualize the first image as below. Notice the differences.
- Look at the data structure of the image I such as its size, type. . . .
- Visualize the green component of the image. Is it different from the red one? What is the most contrasted color component? Why?
- Enumerate some digital image file formats. What are their main differences? Try to write images with the JPEG file format with different compression ratios (0, 50 and 100), as while as the lossless compression, and compare.

See imwrite.

2 Color quantization

Color quantization is a process that reduces the number of distinct colors used in an image, usually with the intention that the resulting image should be as visually similar as possible to the original image. In principle, a color image is usually quantized with 8 bits (i.e. 256 gray levels) for each color component.

- By using the gray level image 'muscle', reduce the number of gray levels to 128, 64, 32, and visualize the different resulting images.
- Compute the different image histograms and compare.

3 Image histogram

An image histogram represents the gray level distribution in a digital image. The histogram corresponds to the number of pixels for each gray level. The MATLAB® function that computes the histogram of any gray level image has the following prototype:

```
function h = histogram(I)
```

The python function has the following prototype:

```
def histogram(I):
```

Compute and visualize the histogram of the image 'muscle.jpg'.

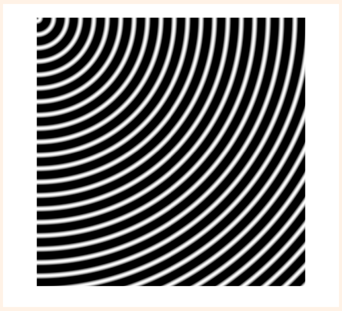
4 Linear mapping of the image intensities

The gray level range of the image 'cellules_cornee.jpg' can be enhanced by a linear mapping such that the minimum (resp. maximum) gray level value of the resulting image is 0 (resp. 255). Mathematically, it consists in finding a function $f(x) = ax + b$ such that $f(min) = 0$ and $f(max) = 255$.

- Load the image and find its extremal gray level values.
- Adjust the intensities by a linear mapping into $[0, 255]$.
- Visualize the resulting image and its histogram.

5 Aliasing effect

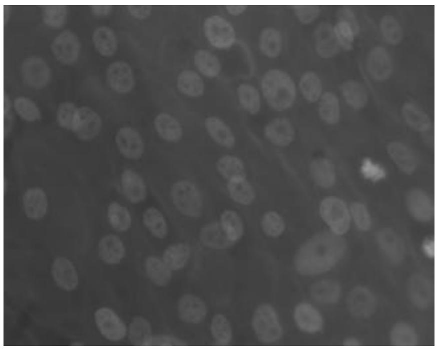
- Create an image (as below) that contains rings as sinusoids. The function takes two input parameters: the sampling frequency and the signal frequency.

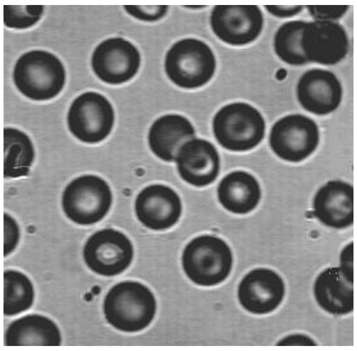


- Look at the influence of the two varying frequencies. What do you observe? Explain the phenomenon from a theoretical point of view.

6 Low-pass filtering

The different processes will be realized on the following images:


(a) osteoblast cells


(b) blood cells

Low-pass filtering aims to smooth the fast intensity variations of the image to be processed.

?

Test the low-pass filters 'mean', 'median', 'min', 'max' and 'gaussian' on the noisy image 'blood cells'.

The MATLAB® functions `imfilter` and `nlfilter` can be employed. Be careful to the function options for border problems. Also, the MATLAB® function `fspecial` enables an operational window to be generated.

The module `scipy.ndimage.filters` contains a lot of classical filters.

Which filter is suitable for the restoration of this image?

7 High-pass filtering

High-pass filtering aims to smooth the low intensity variations of the image to be processed.

- Test the high-pass filters HP on the two initial images in the following way: $HP(f) = f - LP(f)$ where LP is a low-pass filtering (see the previous exercise).
- Test the Laplacian (high-pass) filter on the two initial images with the following convolution mask:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & +8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

8 Derivative filters

Derivative filtering aims to detect the edges (contours) of the image to be processed.

- Test the Prewitt and Sobel derivative filters (corresponding to first order derivatives) on the image 'blood cells' with the use of the following convolution masks:

$$\begin{bmatrix} -1 & 0 & +1 \\ -1 & 0 & +1 \\ -1 & 0 & +1 \end{bmatrix} \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ +1 & +1 & +1 \end{bmatrix}$$

$$\begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix}$$

- Look at the results for the different gradient directions.
- Define an operator taking into account the horizontal and vertical directions.

Remark : the edges could be also detected with the zero-cressings of the Laplacian filtering (corresponding to second order derivatives).

9 Enhancement filtering

Enhancement filtering aims to enhance the contrast or accentuate some specific image characteristics.

?

- Test the enhancement filter E on the image 'osteoblast cells' defined as: $E(f) = f + HP(f)$ where HP is a Laplacian filter (see tutorial about enhancement.).
- Parameterize the previous filter as: $E(f) = \alpha f + HP(f)$, where $\alpha \in \mathbb{R}$.

10 Open question

Find an image filter for enhancing the gray level range of the image 'osteoblast cells'.