## Image processing

An image is a single picture which represents something; it's interesting because it contains a lot of information. Image processing involves changing the nature of an image to improve this information for human interpretation and to render it more suitable for autonomous machine perception.

There's no single best approach: the method must be dictated by the problem at hand.

Image processing aim is obtain skeleton, that is binary image reduced to one pixel tick lines.

Neuron images can be processed using an editor, that allows semi-automatic procedure to obtain skeleton, or  a GUI, that allows more freedom in the choice of parameters. A compromise will be sought between automation and integration of some parameters, because there is no single best approach.

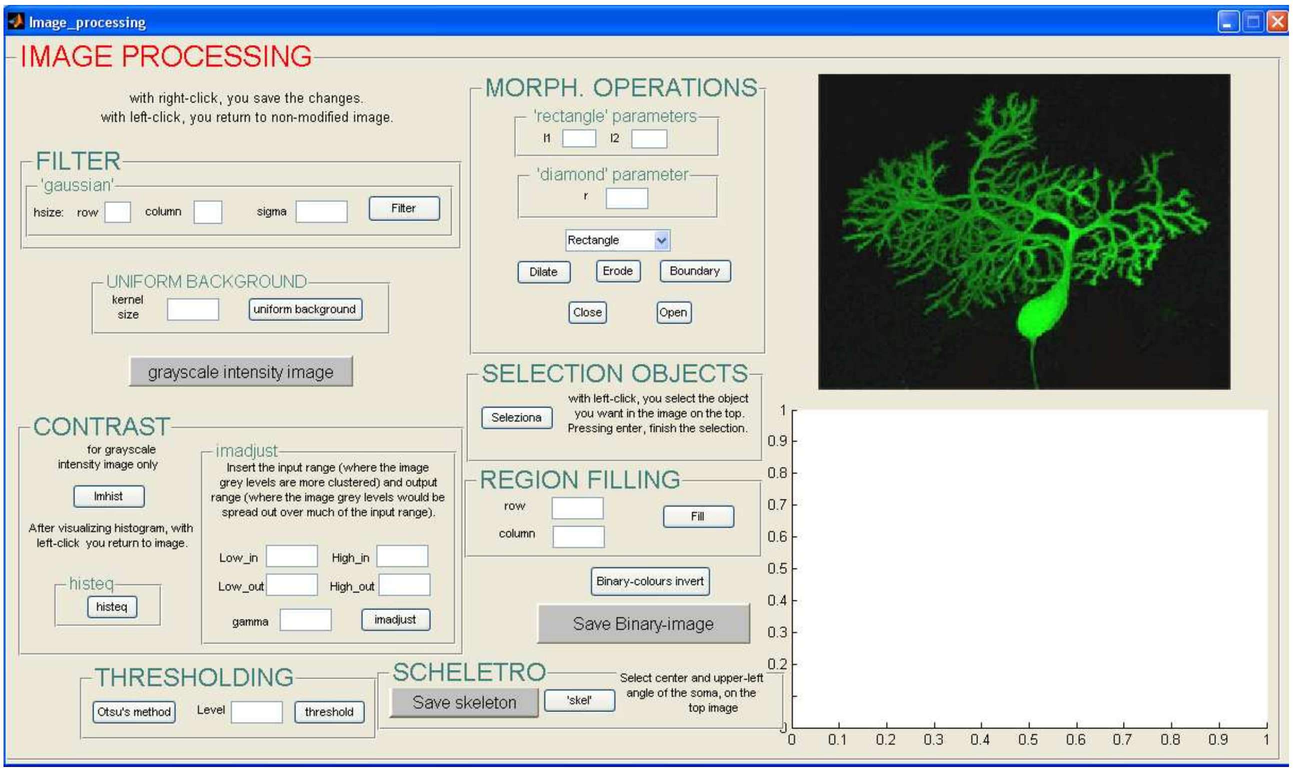
And so,  you can use two different programs to process an image:

- Automatic program
- GUI, for changing some parameters

There are some differences between the editor and the GUI.

First, there are some algorithm implemented only in the GUI: in fact, they are used only for poorly contrasted images.

Then, there are some difference about inserting parameters. In fact, if it's necessary, in the editor, program execution stops and is displayed a message box, which gives the user information about parameters to be included, and then is opened an input dialog box, that allows user to insert them. Instead, in the GUI there are fields, naming edit,  that enable users to enter some parameters. If the user doesn't enter anything, default parameters are automatically used. Dwelling with the mouse cursor on the edit that make up the GUI, appear a label, that describe parameter function, and any default value.



with right-click, you save the changes.
with left-click, you return to non-modified image.

for grayscale intensity image only

After visualizing histogram, with left-click you return to image.

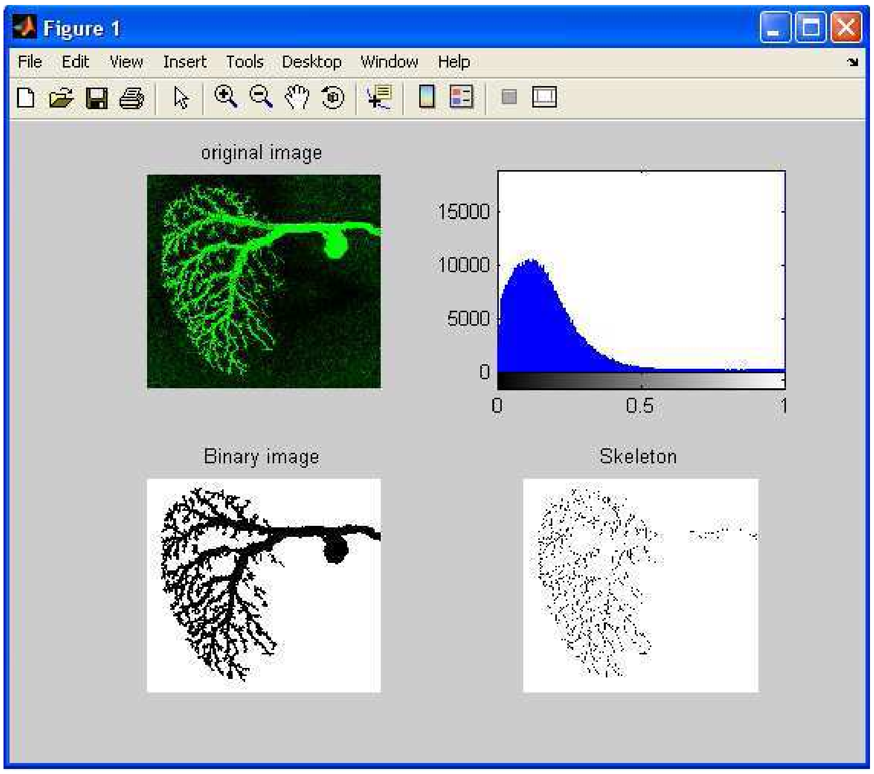
Insert the input range (where the image grey levels are more clustered) and output range (where the image grey levels would be spread out over much of the input range).

with left-click, you select the object you want in the image on the top. Pressing enter, finish the selection.

Select center and upper-left angle of the soma, on the top image

As shown in the figure, there are some panels in the GUI, and each of these is related to one or more algorithms. There are also two panels for displaying photographs: above, it represents image before applying the algorithm, and, below, modified image. So the user, by comparing the two pictures, can decide whether the performed operation has led to optimal image for further processing. For saving changed image, the user will press right mouse button; otherwise, he presses left mouse button.

Instead, in the editor is displayed a window, where the user can see original image and the modified image, aided by the possibility of creating subplot in Matlab. At the end of elaboration, there is a single box representing the original image, the histogram after enhancing the contrast, the binary image and the skeleton, as shown in the next figure . Besides, in the editor, the intermediate images are automatically saved, so there is no possibility of changing, during the processing,  the execution sequence of the algorithms, or of conducting them several times with different parameters.



Finally, in the GUI the user can save binary image and skeleton clicking on “Save Binary image” and “Save skeleton” buttons; in the editor, at the end of processing, first are displayed in different boxes the binary image and the skeleton, then is displayed an other box that ask the user whether to save or not. In both of GUI and the editor, binary image and skeleton are automatically saved in the same folder of the original image.

Both in GUI and editor, you can have information about the pixel in an image (that are pixel value and pixel coordinates) that the cursor is positioned over.

Now, we are going to consider each algorithm; there are some of this implemented only in the GUI: from time to time we are going to specify if the algorithm is also implemented in the editor.

## FILTER

May be convenient to apply a slight gradient image: in fact, we want to minimize small variations in colours and to prevent the detection of nonexistent and unwanted contours: hence has been implemented a Gaussian filter, that is a low pass filter. This algorithm is implemented in both editor and GUI. Inserted in the GUI edits mask size e standard deviation, that model transfer function and define filter area of influence, the user can click on “Filter” button. In the editor, are used default parameters. The algorithm starts: it previously converts image from RGB to YCbCr  space, to isolate luminance component, that is the first matrix plane. Then, it applies low pass filter only on this component, and it reconstructs image by combining the new luminance value with the second and third matrix component, that are unchanged. Lastly, it reports all in RGB space.

## UNIFORM BACKGROUND

Sometimes, can be useful to remove impurities in the image, due to the colture or to the acquisition, so is implemented this algorithm only in the GUI. The operation consists in two phases:

- Estimating brightness of the background: the algorithm performs an opening operation, with a very large structuring element; the result is a matrix, called background, that represents background colour;
- Creating a more uniform background: the algorithm calculates the difference between the starting image and the background.

So we can obtain a more uniform background, and its colour is crearly different from the objects image.

## GRAYSCALE CONVERSION

After running algorithms on truecolor image, it’s possible to convert RGB image to grayscale intensity image. Such the image can therefore be more efficient in term of storage: in fact, it  reduce matrix size, because it is changed from three of two dimensions. Besides, grayscale image can be improved in term of contrast, spreading out its histogram. Obviously, this function is implemented both GUI and editor.

**ENHANCING CONTRAST**

Clicking on “imhist” button in the GUI panel CONTRAST, user can display histogram of image grey levels; that is a graph indicating the number of times each grey level occurs in the image. In the editor, program execution stops and it is displayed the histogram.

There are two ways of enhancing contrast:

- Histogram stretching: the user can stretch the grey levels in the centre of the range out by applying a piecewise function. A Matlab function is used, that maps the intensity values in image f to new values g, such that values between an in range map to an out range. These ranges are inserted by the user, such as a parameter, which describes the shape of the function. This function is also implemented in the editor;
- Histogram equalization: it is an automatic procedure, that change the histogram to one which is uniform, with AHE method. This algorithm is implemented only in the GUI.

In the GUI, can be useful using both the algorithms, so it is possible to take advantage from the single operations.

**THRESHOLDING**

After enhancing image contrast, is necessary to take segmentation, which refers to the operation of partitioning an image into component parts, or into separate objects. A vital part of image segmentation is thresholding, that allows to obtain binary image. It is very efficient in term of storage, because there are only two possible values for each pixel, 0 and 1, that are logical values. So, value of each pixel is described with a single bit.

There are two ways of finding threshold:

- A grey scale is turned into a binary (black or white) image by first choosing a grey level T in the original image , and then turning every pixel black or white according to whether its grey value is greater than or less than T. Value T is visually sought rolling the mouse over the image. This is very important for poorly contrasted image.

- If it is not possible to obtain a single threshold value which will isolate objects completely, the user can click on button “Otsu method”. This algorithm automatically find threshold with Otsu Method. This is implemented both in the GUI and in the editor.

In both the operations, the result is an image with white objects and black background, if photograph represents neurons expressing GFP, so with objects clearer than background. Otherwise, for image representing neurons from wild-type mice, the user has to do a reversal of colours, because object are darker than background.

**MORPHOLOGICAL OPERATIONS**

This branch of image processing is particularly useful for analyzing shapes in images. Four morphological operations are implemented in the GUI:

- Dilate adds pixel to the boundaries of objects in an image. This functions is based on Minkowsky addition. Morphologically dilating image is very useful to connect parts of the same object, which are separated by thresholding.
- Erode removes pixels on object boundaries. This functions in based on Minkowsky subtraction. Morphologically eroding image is useful to remove unwanted items, produced by thresholding.
- Open erodes an image and then dilates the eroded image using the same structuring element for both operations. It removes small object in an image while preserving the shape and size of larger object in the image.
- Close dilates an image and then erodes the dilated image using the same structuring element for both operations. It tends to smooth an image, fuses narrow breaks and thin gulfs, and eliminates small holes.

Number of pixel added or removed from the objects in an image depends on the size and the shape of the structuring element used to process the image. In the GUI, the user can choose between two structuring element shapes, listed in the pop-up menu:

- Rectangle: it creates a rectangle-shape structuring element. The user can specify mask size in the appropriate edit.
- Diamond: it creates a diamond-shape structuring element. The user can specify distance from the structuring element origin to the points of the diamond in the appropriate editor.

In the GUI, the user can choose the sequence of operations, and the same can be done several time. Instead, in the editor, is only implemented dilatation, with rectangle structuring element, and the user can only decide its size.

## BOUNDARY DETECTION & REGION FILLING

With poorly contrasted images, in the GUI boundary detection can be used. Sometimes, it can be a very useful operation. It was implemented using morphological operations. After inserting mask shape and mask size, user will click on “Boundary” button. The algorithm erodes binary images, and the result is subtracted from the original image. Every pixel in a binary image is represented with a logical value: so can be helpful to use logical operator AND.

To detect boundary image, A&~B operation is defined, where B is the result of eroding the image A with the selected structural element. This logical operation always results in 0, unless both elements (A and B) are 1.

Using diamond shape, user can get the best result.

Working on image bounded by an 8-connected boundary, the user can perform this function to fill up the entire region. With the mouse cursor, the user identifies a pixel p inside the object to be filled up, and inserts its coordinates in the appropriate edits. The logical operation is always A&~B, but now A is p-dilatation, with a structuring element, constituted by a matrix 3*3 of all ones, and B is the bounded image. Iterating this operation, A continues to expand until it fills the whole object.

## SELECT OBJECTS

Images can contain multiple objects, which can be neurons, or in vitro impurities. So can be useful to select only some objects in binary image, isolating these from others. When the user clicks on "Select" button, first is implemented a preliminary operations, called "Labelling". This function labels connected objects in binary image. It returns a label matrix, in which the pixels in the same connected objects are assigned the same integer. An 8-adjacency is chosen, so two objects connected only by a pixel are considered as one.

After this operation, labelled image is displayed, and lets the user select objects he wants using mouse. Using normal button, he clicks on a pixel belonging to the object and so he adds all the pixel with the same label, so all the object. Backspace or Delete removes the previously selected object, pressing Return finishes the selection. No-selected objects become automatically black.

This is a very useful technique. Because it allows to visually identify object to be skeletonized, it is implemented both in the GUI and the editor.

## SKELETONIZATION

As mentioned before, skeleton reproduces the structure of the cell, reduced on one pixel tick. Skeleton must have some important characteristic:

- It faithfully must reproduce the structure of the cell;
- It reduce neuron soma of a single pixel, positioned at the its centre.

In literature, we have found two algorithms that carry out skeletonization:

- Latuejoul's method;
- Thinning algorithm.

The first method uses morphological operations to obtain neuron skeleton, the second one Lam, Seong Whan Lee and Ching Y. Suen algorithm, performed by a Matlab command.

Both the algorithms present some problem. In fact, the first method recognizes soma, with reduction of the whole area into a single pixel, but it is not very efficient in reconstructing neuron structure.

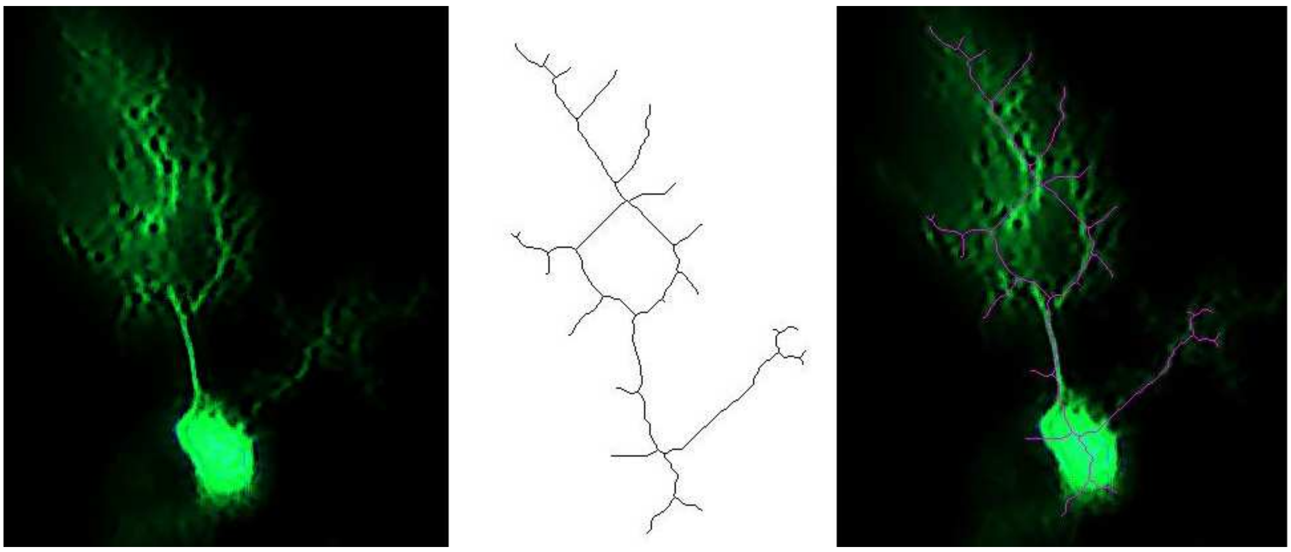
However, the second one is very accurate in the reconstruction of the dendrites, but not so in the soma. So we thought about a synergy between the two algorithms.

In the GUI, clicking on “skel” button, is displayed the input image and the program execution waits for the user to specify a crop rectangle, which must contain soma, with the mouse. In the editor, will be opened a message box, which prompts user to select soma, and automatically opens a window with the input image.

Made selection, the callback operates Lantuejoul’s method on this part. Now, the result is automatically dilated sometimes, to connect the fragments that make up the image. Then, program automatically reintroduces the cropped rectangle in the input image, obviously at the same coordination that has been made on the crop, executes the thinning algorithm, which obtains skeleton from this intermediate image.

On Fig… is represented the result of the synergy between the two algorithms. Are reported original image, skeleton and an overlap between the two.

Is clear that the skeleton meets the specifications given.



Is also clear that the skeleton meets the specifications given, so it is suitable for the extraction of morphological variables.