Segmentation

Introduction

The segmentation of digital images can be performed using a wide variety of methods. In this class we will address three classes of methods that are frequently used in biomedical image segmentation: segmentation methods based on histogram (*thresholding*), methods based on region growing (*region-growing*) and methods based on clustering (*k-means*).

1. Histogram-based segmentation – “Thresholding”

Histogram-based segmentation, or “thresholding”, is a very important approach for image segmentation applications because of its simplicity, being particularly appropriate when there is a good discrimination of the measured characteristics in the image regions to segment. The most frequently used features are the intensity in the case of monochromatic images, and the color components in the case of color images. The major difficulty in using these simple methods lies in the selection of the threshold value(s) to apply, to get the segmented image, which is done by examining the histogram of the image, often using automated methods. When the same threshold is applied to all points of the image we are facing a global segmentation method. If the number of thresholds is just one, the final image is binary and the segmentation process is usually called binarization.
When the image histogram is unimodal or it does not allow a simple separation of the target regions, the use of *thresholding* methods may still be possible if the image is previously divided into regions whose histograms are bimodal and allow the calculation of local thresholds. In other more complicated situations, the threshold value may vary from point to point, being designated as dynamic or adaptive threshold.
When the images have different types of objects that we want to segment we can use multilevel thresholding which is the selection and application of more than one threshold value.
The IPT provides the *graythresh* function that implements an automatic method for the calculation of the threshold, the Otsu method. The syntax of this function is:

```
[T, SM] = graythresh(f)
```

where f is the target image to binarize, and T and MS are, respectively, the threshold (normalized to the range [0, 1]) and a measure of separability resulting from the use of the Otsu method. The image can then be obtained using the *im2bw* function whose syntax is:

```
g = im2bw(f, T)
```

where f and g are, respectively, the original image and the result of the binarization using threshold T.
Multilevel thresholding is implemented through the *multithresh* function whose basic syntax is

```
T = multithresh(f, N)          or        [T, SM] = multithresh(f, N)
```

where N is the number of thresholds and T is a 1xN vector with the threshold values which can be used for converting image f into an image with N + 1 discrete levels (using function *imquantize*).

2. Segmentation based on region growing – “Region growing”

As the name indicates, segmentation methods based on region growing are approaches that group image points, or sub-regions, into larger regions using predefined growth criteria.

The most common approach consists of an initial selection of a set of points that belong to the target objects we want to segment, the seed points, followed by the growth of these segments through the aggregation of neighboring points that verify the growth criteria d. These methods have a very important property which is the combination of similarity characteristics with the requirement for connectivity between the points to aggregate.
The main problem of the methods based on region growing is the definition of the stopping criteria that prevents region overgrowth.
The IPT does not provide any function for implementing segmentation methods based on region growing. However, in this class we will use the *regiongrow* function[1] whose syntax is:

```
[g, NR, SI, TI] = regiongrow(f, S, T)
```

where f is the image to segment, S is a scalar value with the threshold to be used to calculate the set of seed points or an array specifying the set of seed points, and T is a scalar value with the global threshold or an array containing the threshold value to use in each point. If S and T are arrays they must have the same dimension as the image f. The output parameters are the binary image (g), the number of regions (NR), an image containing the seed points (SI) and an image containing the pixels in f that satisfied the threshold test, but before they were processed for connectivity (TI).

3. ***Watersheds***

The morphological approach to the problem of segmentation is implemented in MatLab by function watershed whose syntax is

```
L = watershed (f, conn)
```

where f is the image of the target, conn specifies the type of connectivity to use and L is a matrix with the labels of the regions resulting from segmentation. The application of this function must be preceded by a series of preliminary operations on the image which generally aim to soften the image to reduce noise and local minimum number of image regions. In this way, we intend to minimize over-segmentation that usually characterizes the direct application of segmentation methods based on watersheds. In some situations, the use of watershed function is preceded by other operations such as detection of edges, calculating the distance transform (*bwdist* function) or detection and imposing local minimum regions (*imregionalmin*, *imextendedmin*, *imimposemin*).

4. Clustering (***k-means***)

Clustering is the task of grouping a set of pixels in such a way that pixels in the same group (cluster) are similar (in some set of features) to each other than to those in another group. Matlab provides the kmeans function that performs the partition of points in the data matrix f (N×P) into k clusters. This partition minimizes the sum, over all clusters, of the within-cluster sums of point-to-cluster-centroid distances. Rows of f correspond to points, columns correspond to variables. The syntax of this function is:

```
g = kmeans(f, k)
```

where g is the result of the clustering. When f is a vector, *kmeans* returns an N-by-1 vector g containing the cluster indices of each point. By default, *kmeans* uses squared Euclidean distances.

5. Edge-based segmentation

MatLab does not have specific functions to perform image segmentation based on object contours. To implement this kind of methods the functions studied for edge detection should be used followed by other procedures to improve the link between edge segments in order to get final closed contours.

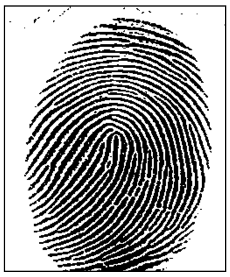
Work proposal

1. (Pen and paper) Consider the image depicted in the figure, which we want to binarize using a region growing algorithm. For this purpose, the detection criteria "intensity value >= 8" and aggregation criteria "intensity >= (mean-1.5)" are used.
a. Highlight the segmented region in the following board, indicating for each point the step of the algorithm in which it was obtained.
b. Discuss the possibility of obtaining the same segment through binarization based on histogram preceded by an adequate pre-processing operation.

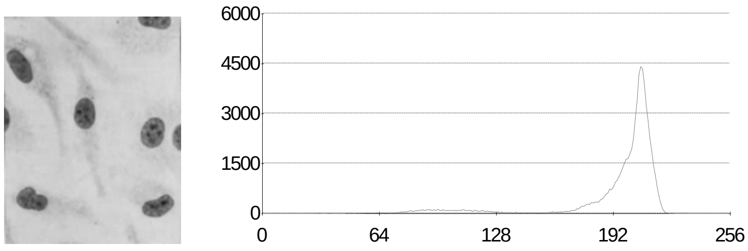
6	6	6	5	5	6	6
6	4	3	4	4	4	7
6	4	5	6	3	4	7
5	3	6	7	6	4	5
5	3	4	8	6	4	5
5	5	4	2	3	4	5
6	5	4	4	4	4	6
7	6	6	5	5	6	7

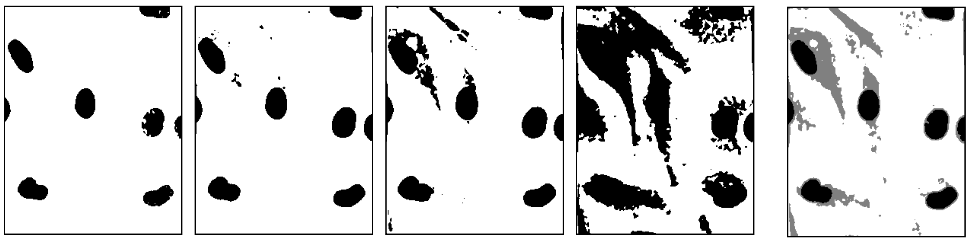
6	6	6	5	5	6	6
6	4	3	4	4	4	7
6	4	5	6	3	4	7
5	3	6	7	6	4	5
5	3	4	8	6	4	5
5	5	4	2	3	4	5
6	5	4	4	4	4	6
7	6	6	5	5	6	7

2. Consider the original image of a fingerprint ("imp_digital.tif") and the image that resulted from its binarization using Otsu's method.
Write a set of instructions in MatLab for:
A. Read and visualize the "imp_digital.tif" image and its histogram;
B. Perform the binarization of the image using a threshold value obtained by histogram observation;
C. Find a new threshold using Otsu's method and obtain the corresponding binary image.
Compare the two threshold values and the two binary images that were obtained.



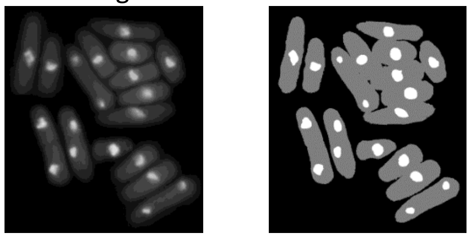
3. The following figures show the "celulas_1.tif" image and its intensity histogram. The original image was segmented using histogram-based binarization ("thresholding") using four different threshold values, T1=128, T2=165, T3=180 and T4=200, giving rise to the four images that are also presented.



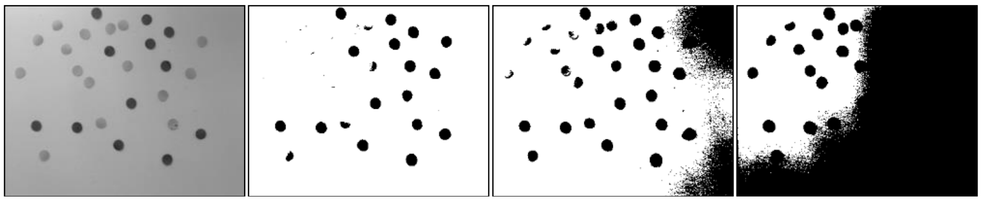

T1=128 T2=165 T3=180 T4=200

Write a set of MatLab instructions for segmenting the "celulas_1.tif" image using multilevel thresholding (two levels), to obtain a segmented image similar to the one shown in the figure. Comment on the obtained values.

4. The "celulas_levedura.tif" image (left image) must be segmented to identify the cells, as well as their nuclei. Use the multilevel thresholding function for obtaining the segmented image that is shown on the right.



5. The image "smarties.tif", presented below, was acquired in deficient light conditions that do not allow segmentation of the small circular objects using a single global threshold. Performing some attempts gave rise to the following results (thresholds values of 135, 150 and 170 on a scale of 0-255 respectively).

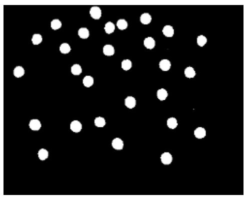


A. Write a set of MatLab instructions to segment the image by partitioning the image into a set of sub-regions and using an appropriate set of local threshold values (selected using the information obtained from the application of the aforementioned global thresholds).

B. Verify if you can get a segmented image similar to that obtained in the previous paragraph but using a set of local thresholds obtained by the Otsu method.

C. An alternative to image partition and subsequent local thresholds definition is the use of dynamic thresholds, consisting in defining a distinct threshold value for each image point. For that purpose, an estimation of the original image background was obtained, as shown in the left figure. The original image can be segmented considering as object points those whose difference (in absolute value) for the respective background is above a threshold value to be set. The background image was obtained through the application of a morphological filter using a structuring element whose size was adjusted to the size of objects present in the image, which do not have a diameter greater than 30 pixels. Write a set of Matlab instructions to perform the suggested segmentation operation.

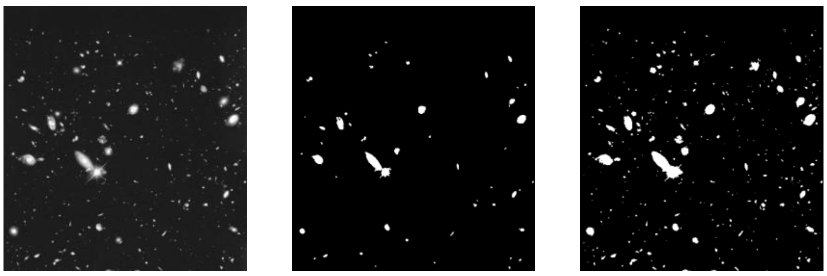
Background


Segmented image

6. In the "Hubble.tif" image (left figure) we aim to segment only the objects with the brightest nucleus to obtain a final image similar to that shown in the center. After verifying that a threshold value equal to 90 (in the range 0-255) allowed the correct detection of the contours of the above mentioned objects, we applied a segmentation method based on histogram with this global threshold value obtaining the image shown on the right. As can be seen, this result is not satisfactory since all image objects are segmented. To solve this problem, we chose to use a segmentation method based on region growing.

A. Analyze the function to perform region growing segmentation whose code is available in the file "regiongrow.m".

B. Write a set of instructions to segment the "Hubble.tif" image and get an image similar to that shown in the center. (To create the seed image for the region growing you may consider that all the objects we want to segment have nuclei with intensity greater than 240).

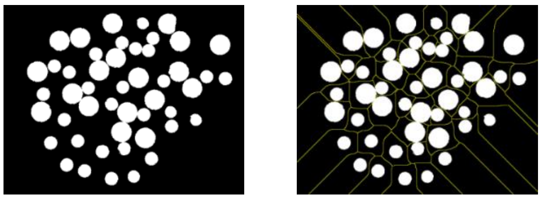


7. Use the K-means clustering for segmenting the "retinograma.tif" image. Use the default parameters and set the random seed to 1 via "rng(1)". After reading the image, assess the segmentation results for different values of *k* using:

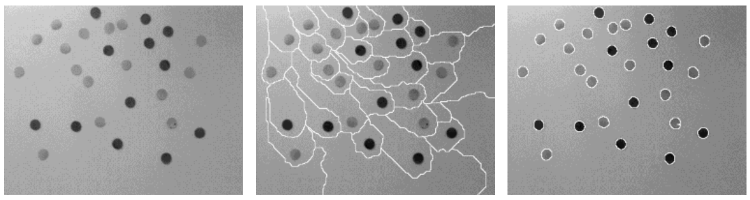
A. The RGB channels as features;

B. Different combinations of 1 and 2 image color channels as input.

8. The “BWCircles.tif" image (left) must be segmented for separating the circular objects that can be observed in the figure. Implement a watershed-based methodology to obtain a result similar to the one shown in the right image.



9. Consider again the "smarties.tif" image shown in the figure on the left and the images that resulted from two segmentation procedures based on watersheds. Write a set of instructions in Matlab for obtaining segmentation results similar to those shown in the following figures.



A. Apply *watershed* directly to the image;
B. Smooth the image via appropriate morphological operations prior to the *watershed*;
C. Use an edge enhancement technique, followed by noise removal, prior to the watershed;

10. Implement the k-means clustering method by hand (i.e., without using the *kmeans* function from MATLAB) for segmenting the image of problem 2.

[1] Digital Image Processing using MatLab, 2nd Ed., Gonzalez, Woods and Eddins, Prentice-Hall, 2009.