# COMPUTER VISION

**EXERCISE 4a: Image segmentation with region growing**

Concepts: Region growing, seed

The code shown below extracts a region of the image by a growing process. This process starts with a seed in the (*x*,*y*) coordinates and adds pixels to the region recursively. Implement a code to:

* Load the *rice.tif* image and initializes the global variables **region**, **media**, **points\_in\_region** and **im1**.
* Permits the user to indicate a seed and initializes, using that seed, the growing process. You can use the **ginput** command for that. *Note: when testing the code, choose the seed in such a way that it falls into a rice grain.*
* Execute the region growing function (**crec\_recur**), and draw the extracted region in the initial image as it is shown in the result image.

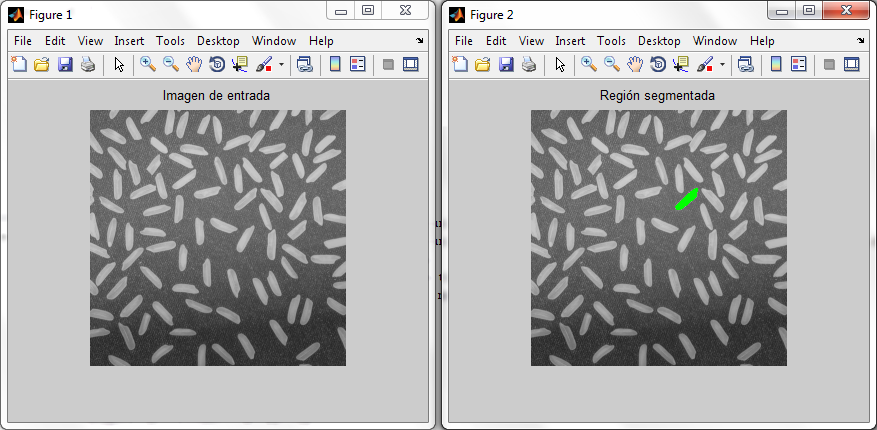
**Code**

|  |
| --- |
| % Esta función realiza de manera recursiva el crecimiento de una región de la imagen  % partiendo de una semilla en (x,y). El crecimiento se hace sobre sus ocho-vecinos.  %  % J. Gonzalez  function crec\_recur(x,y,toler)    % Variables globales para no pasar tantos argumentos a la funcion  global region; % Region segmentada. Es una matriz del tamaño la imagen,con  % todo a cero salvo la region, a uno.  global media; % Media (dinamica) de la region que va creciendo (DOUBLE!).  global points\_in\_region; % Numero de puntos actual de la region segmentada  global im1; % Imagen de entrada en escala de grises    % Comprobacion que no estamos fuera de la imagen  if (x <= 0 | x > size(im1,2) | y <= 0 | y > size(im1,1) )  return;  end  if (region(y,x) == 1) % si esta ya marcado (=1) no hacemos nada y salimos de la funcion  return;  end    if abs(double(im1(y,x)) - media) < toler % Condicion de añadir pixel  region(y,x) = 1;  points\_in\_region = points\_in\_region + 1;  media = (media \* points\_in\_region + double(im1(y,x)))/(points\_in\_region+1); %calculo dinamico de la media    % Recursion sobre los 8-vecinos  crec\_recur(x-1,y+1,toler);  crec\_recur(x-1,y,toler);  crec\_recur(x-1,y-1,toler);  crec\_recur(x,y+1,toler);  crec\_recur(x,y-1,toler);  crec\_recur(x+1,y+1,toler);  crec\_recur(x+1,y,toler);  crec\_recur(x+1,y-1,toler);  end    return; |

**Matlab functions:**

|  |  |
| --- | --- |
| **global var** | Declares the **var** variable as global, i.e. it can be used at any point of your code that it is also declared as global. |
| **[x,y] = ginput(1)** | Permits us to get a point from the active axes and returns its coordinates in the **x** and **y** variables. |

**Results**



**EXERCISE 4b: Image segmentation using K-means**

Concepts: K-means

The K-means algorithm is used for clustering, since it permits us to group data (in our case, pixels) relying on a certain criteria, for example, a similar intensity level. In the code example below it is illustrated how to use the Matlab function **kmeans** employing as features the image pixel intensities for the segmentation of the image *torre\_monica.jpg* (in grayscale) into two clusters.

* Execute the code. What’s happening?
* Change the number of iterations of the *for* loop until the **kmeans** algorithm reaches convergence. How many iterations are needed?
* Now, modify the code to segment objects in the RGB 3D space, in other words, the feature of each pixel is its RGB color. *Pay special attention to the reshape function; get help with* ***help reshape*** *in Matlab.*
* Compare the resultant segmentation with the one obtained for the same image but in grayscale.

***Note:*** *for more information about* ***kmeans*** *you can check the Matlab help and the lectures’ material.*

Example:

|  |
| --- |
| clear variables  close all  clc  im **=** imread**(**'torre\_monica.jpg'**);** % Load the image  im\_gray **=** rgb2gray**(**im**);** % Convert it to grayscale  figure**,** subplot**(**1**,**2**,**1**),** imshow**(**im\_gray**),** title**(**'Initial image'**)** % Show it  nPixels **=** prod**(**size**(**im\_gray**));** % Get the number of pixels to ...  data **=** reshape**(**im\_gray**,** nPixels**,** 1**);** % ... reshape the image as a vector  seed1 **=** **[**10**,**10**];** % Set the seeds  seed2 **=** **[**400**,**240**];**  seeds **=** **[**seed1**(**1**)\***size**(**im\_gray**,**2**)+**seed1**(**2**);**seed2**(**1**)\***size**(**im\_gray**,**2**)+**seed2**(**2**)];**  % seeds = repmat(seeds,[1,3]); % Uncomment for the RGB image  **for** i**=**1**:**10  fprintf**(**'Performing k-means segmentation with %d maximum iterations\n'**,**i**);**  pert **=** kmeans**(**double**(**data**),** 2**,**'Start'**,**seeds**,**'MaxIter'**,**i**);** % kmeans!  clus **=** reshape**(**pert**,** size**(**im\_gray**));** %Vector Image back to a matrix  im\_clust **=** uint8**(**255**\*(**clus**-**1**)/(**max**(**max**(**clus**))-**1**));** % Clusters are 1 and 2  subplot**(**1**,**2**,**2**),** imshow**(**im\_clust**),** title**(**'Segmented image'**)** % Show results  pause**;**  **end** |

**Matlab functions:**

|  |  |
| --- | --- |
| **[belonging]=kmeans(datos, *K*)** | Applies the kmeans algorith to the set of **data** and tries to group them into **k** clusters. The **belonging** vector contains, for each element in the **data** vector, the cluster to which it belongs. |

**Results**

**EXERCISE 4c: Image segmentation with EM (Expectation Maximization)**

Concepts: EM

The EM algorithm assigns to each region or cluster a Gaussian probability distribution and enables us to group data (in our case, pixels) by maximizing the probability of the data (pixels) belonging to each region.

* Segment the *torre\_monica.jpg* into two regions using **segmentation\_em**, available as material for the course, using a different number of algorithm iterations. Concretely try *numIter=2*, *numIter =5* and *numIter =10.*
* Interpret the probability density functions of the obtained regions for the different executions. *Note: these densities are automatically plotted by the* ***segmentation\_em*** *function.*
* Compare the results with the output of the K-means algorithm working with the grayscale image and justify the differences.

Example:

|  |
| --- |
| clear variables  clc  % Prepare the data  num\_seg **=** 2**;**  im **=** imread**(**'torre\_monica.jpg'**);**  im **=** rgb2gray**(**im**);**  im **=** im2double**(**im**);**  numIter**=** 2**;**  clus **=** segmentation\_em**(**im**,** num\_seg**,** numIter**);** %Image with segm. result  im\_clust **=** uint8**(**255**\*(**clus**-**1**)/(**max**(**max**(**clus**))-**1**));**    % Show the figure  figure  subplot**(**1**,**2**,**1**);**  imshow**(**im**);**  title**(**'Imagen original'**);**  % Show the results  subplot**(**1**,**2**,**2**);**  imshow**(**im\_clust**);**  title**(**'Segmentacion EM'**);** |

**Matlab functions:**

|  |  |
| --- | --- |
| **Segmented\_im = segmentation\_em(im, K)** | Executes the EM algorithm to segment the **im** image (grayscale) into a number of regions **K** using the pixels intensity. Its output is an image with the same size which elements indicate the region to which they belong. The input image **im** must be in double (0 to 1) format. |