ENGG 111 - Homework 6 - Region Growing and Watershed Transforms

Preliminary

This assignment should not be very difficult, although it may take some programming effort. If you feel you are having difficulties, please see me so I can help you. Do not wait too long to get started, the programming work while manageable may present some algorithmic difficulties.

Region Growing

In this assignment I would like you to program a "strictly correct" version of the region growing algorithm. Write it as a function:

imOut=regionGrow(imIn, seeds);

which will return an image labeled with as many regions as there are seeds. The image imIn should be an intensity (gray, type double) image. The seed array has two columns showing the row and column coordinates of the seed points. You are welcome to use my example code as a starting point.

I would like you to try your function on the following images: MRCoronal1.dcm (a DICOM single slice) and "blobs.tif" (a micro photograph). You may present additional images if you like to brag about how good the algorithm is. In this demonstration you may pick your seed points manually. For the DICOM images you may pick seed points corresponding to the "outside" then one to the "brain" and one to the "skull" areas. For the blobs you may want to define a seed for each recognizable blobs and one for the background.

Remember, the strict version of the region growing algorithm consists of:

a) given existing regions or seed points - Grow to define the contiguous regions of candidate pixels for addition

b) for each region, pick that pixel (or multiple pixels if several have the same value) that is closest to that regions's mean

c) recompute the region means

d) If there are still unassigned pixels go to a)

Remember also to make provisions to avoid regions taking pixels from each other.

Next modify your function so that if it is given a scalar instead of an array of seeds, that scalar number will represent the number of randomly placed seeds to use to initiate the region growing algorithm. In matlab you may use any number of methods to come up with 100 random [r c] pairs. Note that they should be integer and be in the range 1 to #rows and 1 to #cols respectively. Make provisions to avoid seed points that either are duplicates or that are contiguous. Next, I would like you to use your function with 100 randomly selected seed points.

Performing region growing with 100 seed points will give far too many regions to be useful. What you should do next is to merge contiguous regions based on the similarity of their mean values. Merge regions based on which are most similar first and continue doing this until you have no more than 6 regions (for blobs.tif) or 3 regions for "00010064", similar to the 1st part of this exercise. Remember to re-compute means after any merger.

The Watershed Transform

A noisy image such as an MRI slice does not lend itself to a very useful segmentation using the watershed transformation because it produces too many regions. We saw that smoothing alone can take care of that problem in part, but not completely (usually). I suggested one remedy which was to merge contiguous basins on the basis of similarity. In this assignment you will explore that idea. In order to complete this assignment I strongly suggest that you read carefully the help files on the functions provided by Matlab for the watershed transform.

Using that same test image (MRCoronal1.dcm), I invite you to explore the following procedure:

a) smooth the image using a Gaussian for which you select σ and n

b) apply the watershed transform

c) Using the "basins" (the different regions resulting from the watershed transform) merge the different regions based on some "similarity" criterion in the pixel values from the original image.

I understand that these instructions are short on specific implementation details. I would like you to write the above series of steps in a Matlab function. Inside your function you can use any built in Matlab function. If you write this code with parameters (i.e. σ and n for the smoothing filter size) you will be able to try different parameter values and see what works well. You will need to come up with a "good" (workable) smoothing filter and a "good" way to decide when to merge and when not to merge two contiguous regions.

I suggest that you try the following criteria for merging contiguous regions:

1) mean value for the region

2) the bottom (lowest pixel value for that region)

Note that it is best to merge first those regions that are contiguous and most similar by your selected criterion. It is also critical that any time you merge two regions you recompute the similarity criterion (depth or mean).

I will consider a good result to be a procedure that separates the main features of the image properly: brain, possibly brain regions (i.e. cerebellum vs cerebrum), bone/scalp/fat etc.