Segmenting with ITK (II)
Some notes

© 2020 Alejandro J. León Salas

February 15, 2020

Contents

1. What is this session about?

In this session we are going to practice different techniques provided by ITK for doing segmentation over images, mainly *region growing* and *watershed* segmentation. We will be referring a lot to the chapter on ‘Segmentation’ you can find in the ITK *guide2* [1].

The image segmentation process is not a trivial one. There is no single approach that can generally solve the problem of segmentation for the large variety of image modalities existing today. To obtain a satisfactory final result we have to customize combination of components and parameters in the pipeline. The parameters of these components are tuned for the characteristics of the image modality used as input and the features of the anatomical structure to be segmented.

1.1. Learning outcomes

- To understand the region growing segmentation technique by exploring how its ITK implementation works on some examples.
- To understand the watershed segmentation technique by exploring how its ITK implementation works on some examples.
- To know how to use Mathematical Morphology operators to refine the segmentation results.

2. Region growing

The basic approach of a region growing algorithm is to start from a seed region (typically one or more pixels) that the user consider inside the region to be segmented. Region growing algorithms vary depending on the criteria used to decide whether a pixel should be included in the region or not, the type of connectivity used to determine neighbors, and the strategy used to visit pixels included in the neighborhood. The ITK toolkit provides several implementations of the region growing approach[1].

2.1. Connected Threshold

The simplest criterion for including pixels in a growing region is to evaluate whether the candidate pixel’s intensity value is inside a specific interval or not. The interval of intensity values is provided by the user as lower and upper threshold values. Those pixels whose intensities are inside the interval are included in the region.

Noise present in the image could reduce the capacity of this filter for growing large regions. When faced with noisy images, it is usually convenient to pre-process the image by using an edge-preserving smoothing filter. We will use `CurvatureFlowImageFilter` which requires a couple of parameters. We will use $NumberOfIterations = 5$ and $TimeStep = 0.125$ for a first try but these values may have to be adjusted depending on the amount of noise present in the input image. Read the code in `ConnectedThresholdImageFilter.cxx` to know how to combine these filters.

This program uses several filters in a pipeline, a reader, a smoothing filter (`CurvatureFlowImageFilter`), a region growing segmentation filter (`ConnectedThresholdImageFilter`), and a filter for type casting (from `float` to `unsigned char`). For using `ConnectedThresholdImageFilter` we must establish two parameters, `lowerThreshold` (`SetLower()`) and `upperThreshold` (`SetUpper()`), together with a seed (`index`) included in the region of the image we want to segment (`SetSeed()`). This is the main difference from a thresholding filter because we are selecting the region we are interested in segmenting from the rest of the image, hence we are specifying information on the geometric domain, not just in the intensity domain.

[1] Look for more information in section ‘Region Growing’ in the ITK *guide2*.

The output of `ConnectedThresholdImageFilter` is a binary image with zero-value pixels everywhere except on the extracted region. The intensity value set inside the region is selected with the method `SetReplaceValue()` .

2.2. Neighborhood Connected

This is a variant of the filter we have shown in the section above. The `ConnectedThresholdImageFilter` only considers the value of the current pixel in order to check whether this value is within the interval $[lowerThreshold, upperThreshold]$ or not, and if so, the pixel is included in the region. `NeighborhoodConnectedImageFilter` considers a user-defined neighborhood surrounding the pixel, requiring that the intensity of each neighbor be within the interval for it is included.

The reason for considering the neighborhood intensities instead of just the current pixel intensity is due to small structures are less likely to be accepted in the region during the execution of the region growing algorithm. The operation of this filter is equivalent to applying `ConnectedThresholdImageFilter` followed by *mathematical morphology erosion* using a structuring element of the same shape as the neighborhood provided to the `NeighborhoodConnectedImageFilter` . We will practice mathemathical morphology in the corresponding section below. Try the program `NeighborhoodConnectedImageFilter.cxx` to understand this filter.

As in `ConnectedThresholdImageFilter` you must pass two parameteres, upper and lower threshold, together with the seed and an output value for the segmented region. However, for this new filter you must include the neighborhood size as a parameter. This is done using the method `SetRadius()` which represents the number of pixels along each dimension from the current pixel. Remember that the neighborhood in ITK is calculated as a rectangular region with $2 \times radius + 1$. For example, if you select a radius $[2, 2]$, you will get a neighborhood of 5×5 pixels.

2.3. Confidence Connected

The criterion used by the `ConfidenceConnectedImageFilter` for growing the region is based on statistics calculations made over the pixels values of the current region.

First, the algorithm computes the mean and standard deviation of intensity values for all the pixels currently included in the region. A user-provided factor is used to multiply the standard deviation and define a range around the mean. As usual in region growing, neighbor pixels whose intensity values fall inside the range are accepted and included in the region. When no more neighbor pixels are found that satisfy the criterion, the algorithm is considered to have finished its first iteration. The following equation shows the interval considered for including a pixel into de region:

$$I(pixel) \in [m - f\sigma, m + f\sigma]$$

where m represents the mean value and σ is de standard deviation of intensity values. At that point, the mean and standard deviation of the intensity levels are recomputed using all the pixels currently included in the region that just have been calculated. This mean and standard deviation defines a new intensity range that is used to visit current region neighbors and evaluate whether their intensity falls inside the range. This iterative process is repeated until no more pixels are added or the maximum number of iterations is reached.

Two parameters are required to use `ConfidenceConnectedImageFilter` , f and number of iterations. The factor f defines how large the range of intensities will be. Small values of the multiplier will restrict the inclusion of pixels to those having very similar intensities to those in the current region. Larger values of f will result in more generous growth of the region. Values that are too large will cause the region to grow into neighboring regions which may belong to separate anatomical structures.

The number of iterations is specified based on the homogeneity of the intensities of the anatomical structure to be segmented. Highly homogeneous regions may only require a couple of iterations. MRI images with inhomogeneous fields may require more iterations. Try out to run the program `ConfidenceConnected.cxx` .

Task 1

Write a program that reads an image from a file and displays the result of applying the following filters to the input image: `ConnectedThresholdImageFilter` , `NeighborhoodConnectedImageFilter` , and `ConfidenceConnectedImageFilter` . The idea is to obtain several results by combining the free parameters of each filter, and by using or not a noise reduction filter over an input image before applying each one of these filters. The idea is to learn how the different filter parameters affect the segmentation result and observe how the application, or not, of a noise reduction filter previously in the pipeline affects the segmentation result. Try this program with the input images: `BrainProtonDensitySlice256x256.png`, `coronaryAngiogram.png`, `saltAndPepperNoise1.jpg`, and `gaussianNoise.jpg` .

3. Watershed segmentation

You can read the subsection 'Overview' in the ITK *guide2* (included in section 'Segmentation based on Watersheds') for remembering the basis of watershed segmentation, but we are going to focus on the use of the ITK Watershed filter illustrated in section 'Using the ITK Watershed Filter' by using the code in the file `WatershedSegmentation1.cxx` . Remember that watershed approach does not produce a single image segmentation, but rather a *hierarchy of segmentations*.

The strategy of watershed segmentation is to treat an image f as a *height function*, i.e., the surface formed by graphing f as a function of its independent parameters, $\mathbf{x} \in U$. The image f is often not the original input data, but is derived from that data through some filtering, graded (or fuzzy) feature extraction, or fusion of *feature maps* from different sources. The assumption is that higher values of f indicate the presence of *boundaries* in the original data. Figure 1 shows the result 1(b) of applying a watershed segmentation over a gradient magnitude image 1(a) resulting from a feature extraction filter on a MRI image[2].

Watersheds may therefore be considered as a final or intermediate step in a hybrid segmentation method, where the initial segmentation is the generation of the *edge feature map*, i.e. an edge detection process is previously carried out. Typically, the best results are obtained by preprocessing the original image with an edge-preserving diffusion filter, such as one of the anisotropic diffusion filters, or the bilateral image filter. The height function used as input should be created such that higher positive values correspond to object boundaries. A suitable height function for many applications can be generated as the *gradient magnitude* of the image to be segmented.

[2] By The original uploader was Coupriec at English Wikipedia. - Transferred from en.wikipedia to Commons by Adelpine using CommonsHelper., CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=14768570 and https://commons.wikimedia.org/w/index.php?curid=14763794

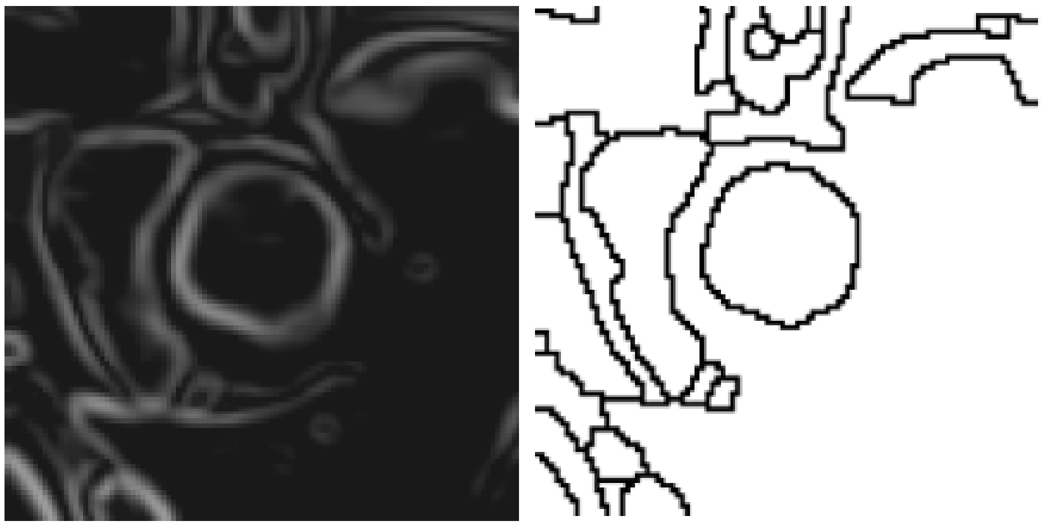
(a) Gradient magnitude image. (b) Resulting watershed segmentation.

Figure 1: A watershed segmentation (b) of a gradient magnitude image (a).

The drawback of watershed segmentation is that it produces a region for each local minimum and there are lots of local minima in common medical images. This fact leads to a oversegmented image. To reduce this drawback, we can establish a minimum *watershed depth*. The watershed depth is the difference in height between the watershed minimum and the lowest boundary point. In other words, it is the maximum depth of water a region could hold without flowing into any of its neighbors (See Figure 2).

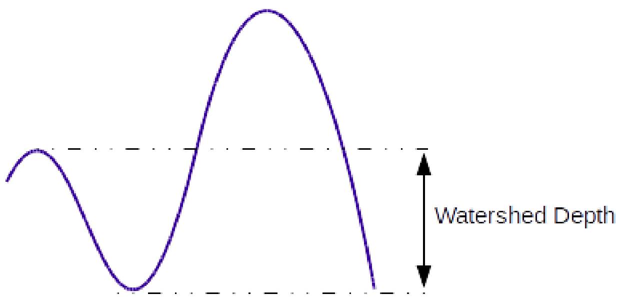

Figure 2: The difference in height between the watershed minimum and the lowest boundary point (watershed depth) is shown in 1D.

Thus, a watershed segmentation can sequentially combine watersheds whose depths fall below the minimum until all of the watersheds are of sufficient depth. This depth measurement can be combined with other *saliency measurements*, such as size. The result is a segmentation containing regions whose boundaries and size are significant. Because the merging process is sequential, it produces a hierarchy of regions (merge tree).

Task 2

Now it's time to put on with real stuff! The code in `WatershedSegmentation1.cxx` shows how to use `WatershedImageFilter`. I would strongly recommend to read ITK documentation of `itk::WatershedImageFilter` , especially the sections 'Description of the input to this filter' and 'Some notes on filter parameters'. Run this

4. Mathematical Morphology

Mathematical morphology filters are an important tool in image processing. ITK provides us with filters that operate on binary images and filters that operate on grayscale images. Read the section on “Mathematical Morphology” in the ITK *guide2* and try the code examples.

Exercise

Your exercise is just to use the programs you have written in **Task1** (region growing segmentation) and **Task2** (watershed segmentation) for writing a report. Take as input several MRI images from `data` and try to segment a couple of structures by using the two programs you have written with different parameter settings for the filters. Finally, try to use a mathematical morphology binary filter to correct some parts of a segmention result. Write a report on the different results you have obtained concerning the different types of images you have provided to your program, the different combinations of parameters and noise filtering on/off, and mathematical morphology on/off.

References

[1] Johnson, H. J., McCormick, M. M., and Ibanez, L. *The ITK Software Guide Book 2: Design and Functionality-Volume 2 4th ed.*, 2017.