COMP2005 Laboratory Sheet 7: Segmentation

1. Region Growing

Region Growing is a bottom-up approach. Starting from the seed point, the region is grown by adding neighbouring pixels if they satisfy a pre-defined criterion. In this case, the criterion is if the difference in pixel intensity is within the threshold selected. A Python implementation of a region growing algorithm is available on Moodle (*regiongrow.py*). **Please inspect and understand it.**

regiongrow(img, seed, threshold)

- *img*: the input image
- *seed*: seed point selected by your mouse
- *threshold*: threshold for the similarity criteria between pixels

Use *regiongrow* to segment out the cells in the image *defective_weld.tif.* Adjust the *threshold* and *seed* parameters as necessary.

2. Split and Merge

Split & Merge is a top-down approach. Starting from the input image, the algorithm first splits the image if the standard deviation of the region is above a certain threshold. To merge regions, the regions must fulfil two criteria: the regions are adjacent, and the regions fulfil the merge criteria. A Python implementation of the split & merge algorithm is available on Moodle (splitmerge.py). **Please inspect and understand it.**

Initialise an instance of the class:

split_merge = SplitMergeMaster(img, split_thresh, merge_thresh)

- *img*: the input image
- *split_thresh:* the threshold for the split criteria
- *merge_thresh:* the threshold for the merge criteria

Perform split merge segmentation:

masks = split_merge.segmentation()

Visualise the results of the segmentation:

canvas = split_merge.visualization(masks)

Use the above to segment the tree in a greyscale version of the image *Tree.tif* provided on Moodle. Adjust the *split_thresh* and *merge_thresh* parameters as necessary.

3. **Watershed**

A watershed is a thin line (at the top of a ridge) that arises from the boundary of a closed region. This is an edge-based segmentation approach. Python provides an implementation of the watershed algorithm through the function *watershed*, but some steps must be done first. OpenCV has provided a concise explanation of how to implement the watershed algorithm. **Please read and understand** the steps: https://docs.opencv.org/4.x/d3/db4/tutorial_py_watershed.html.

Following OpenCV's implementation, modify the parameters to segment the image *blob.tif* into regions. Why do you think this works quite well?

Now apply the method to a greyscale version of the *Tree.tif* image, then try to improve your segmentation by filtering the image first.

4. **Expected Results**

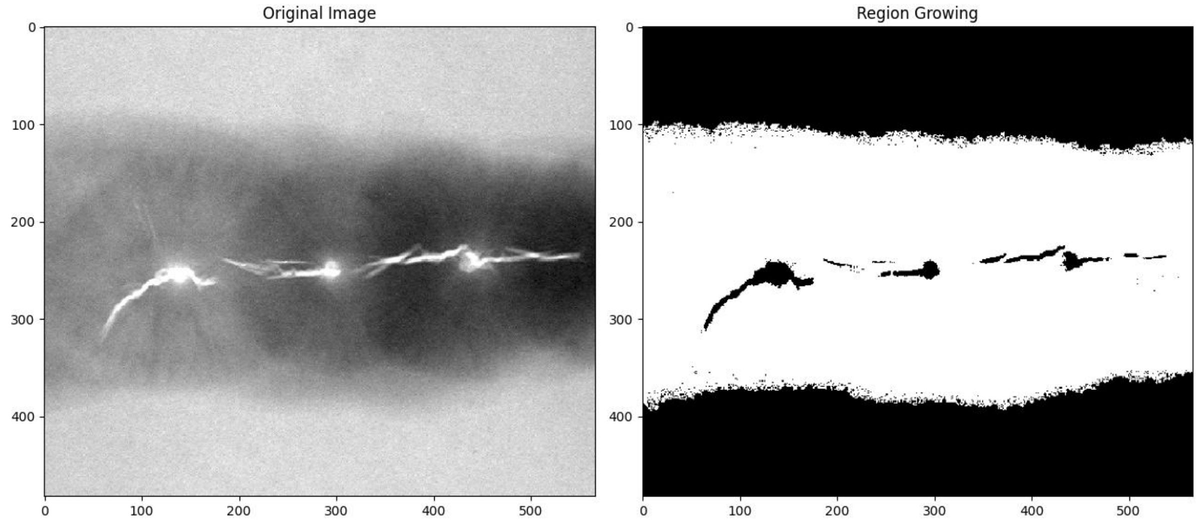

Figure 1: Region Growing

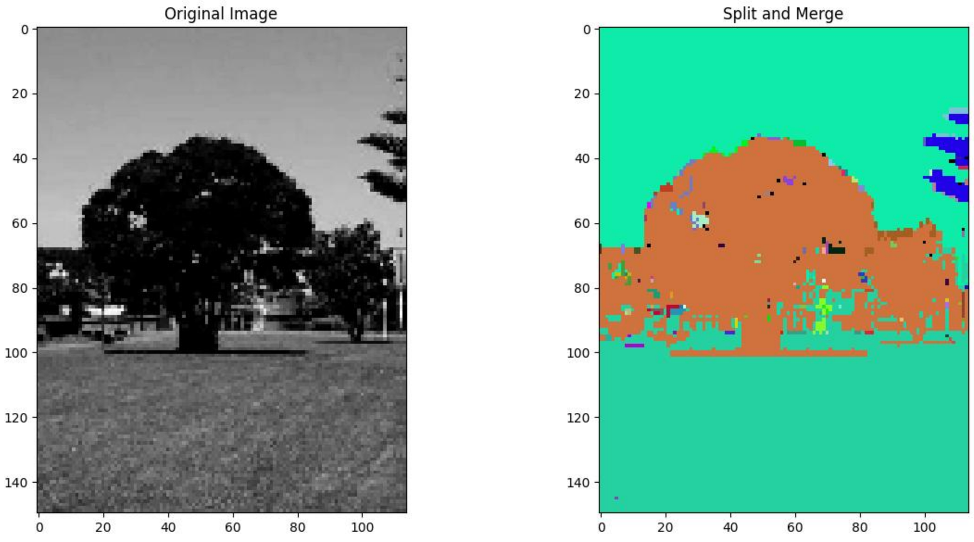

Figure 2: Split and Merge

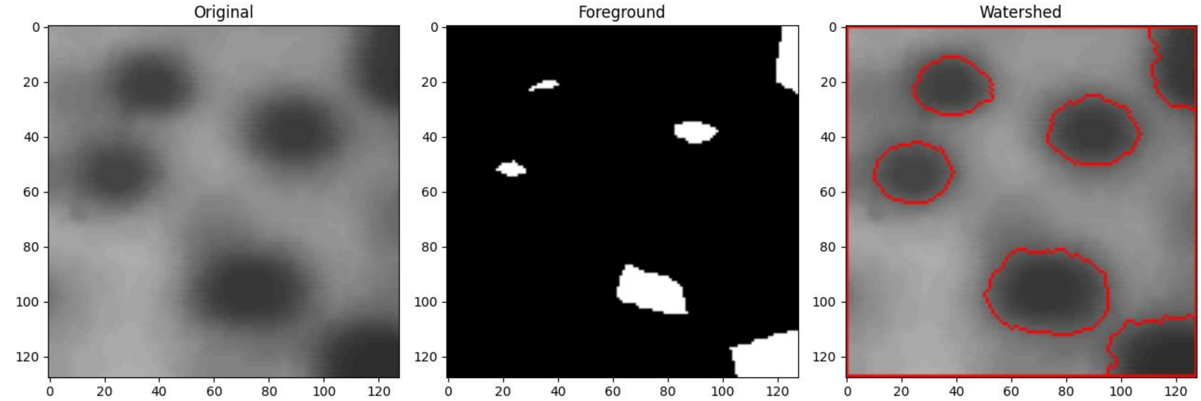

Figure 3: Watershed