Homework 4: Detection & Segmentation

Zach Brown - Oct. 19, 2023 - DIP 4650

Abstract

This homework focused on the implementation of the Laplacian of Gaussian filter for blob detection and the K-means clustering method to segment the image. The results obtained show that both of these requirements aren't just met, they are exceeded.

Introduction

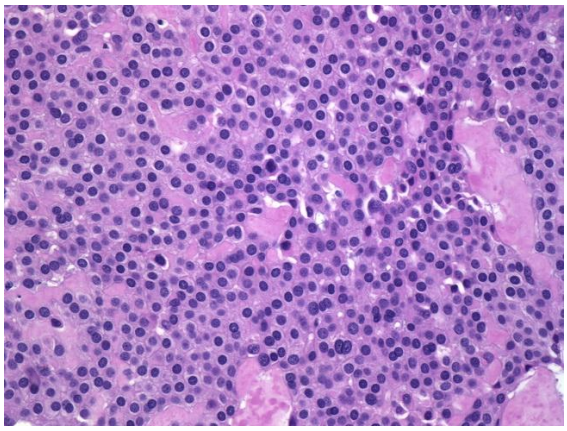
The first task was to preprocess the image using either Gaussian or Median filter to smooth the input images and to remove noise. The second task was to detect blobs in the image. We first were instructed to convert the image to grayscale, and then use the scale-space Laplacian of Gaussian (LoG) filter we learned about in lecture. The third and final task was to perform image segmentation using K-means clustering. For this, we had to use the K-means clustering method we learned about in lecture, and then use two different feature sets to show our results.

Experiments

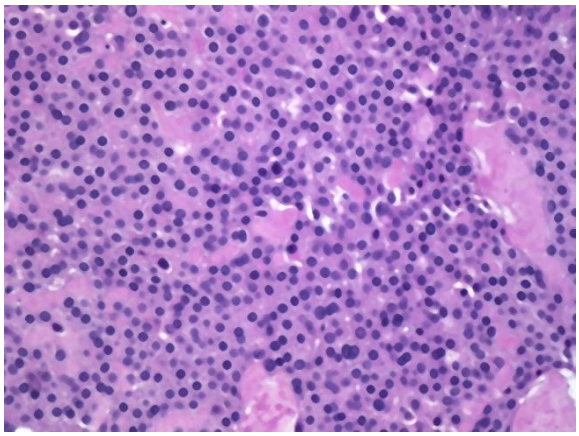
Task 1 - Preprocessing

This step wasn't entirely necessary; the images are already very clear, but it is a good habit to always preprocess when designing an image processing pipeline. I will be processing my images with a median filter of kernel size 5. I chose this filter and size for a couple of reasons:

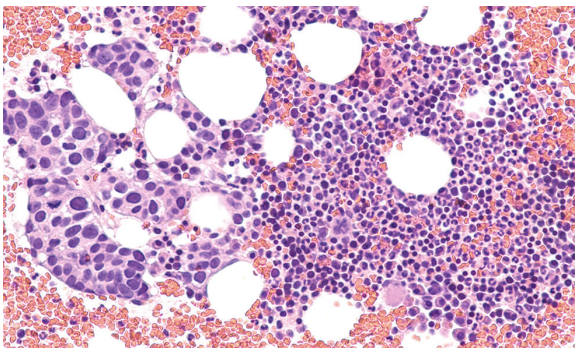
- Noise Removal
- Weak Smoothing



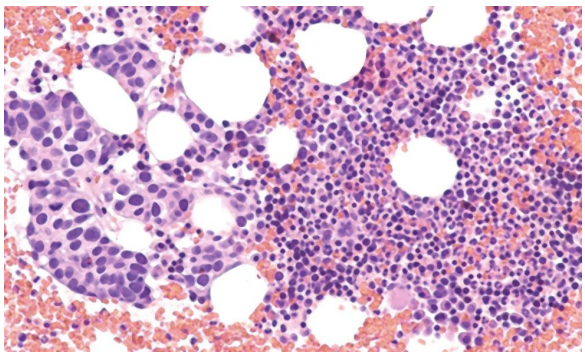
Original Image



Processed



Original Image



Processed

Task 2 - Blob Detection

This step gave me the most trouble by far. I heavily leaned on the work of Hui Kong, Hatice Cinar Akakin, and Sanjay E. Sarma in their paper “A Generalized Laplacian of Gaussian Filter for Blob Detection and Its Applications” which can be found at the following URL: https://ieeexplore.ieee.org/abstract/document/6408211. I implemented a variation of their less general (non-elliptical nor angled) LoG detector which they define as:

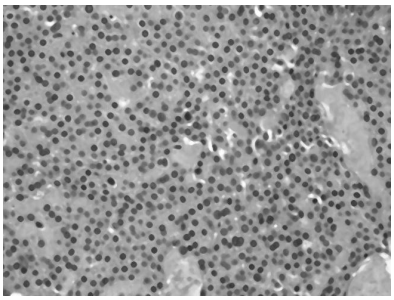
$$\nabla^2 G(x,y) = \frac{x^2+y^2-2\sigma^2}{\pi\sigma^2} exp(-\frac{x^2+y^2}{2\sigma^2})$$

as well as the scaled normalized response they denoted as:

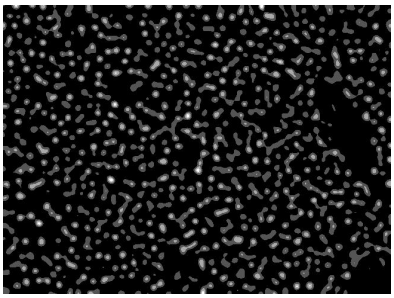
$$L_n = (1 + log(\sigma)^{\alpha})^2 \nabla^2 I(.;\sigma)$$

This scaled normalized response obtained strong results.

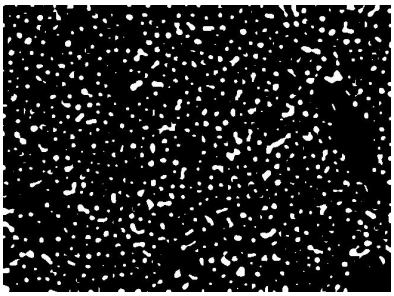
Glomus Tumor (σ=5, threshold=0)



Grayscale

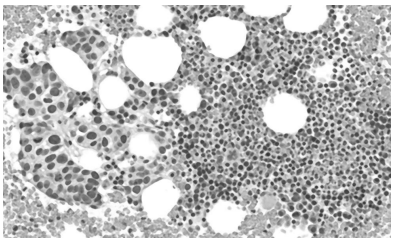


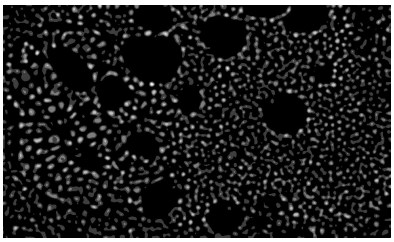
LoG Output

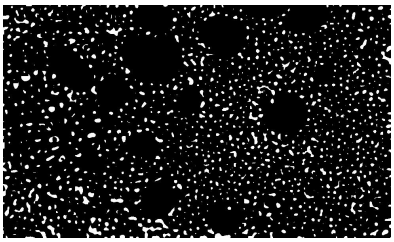


Binary Mask

Metastatic Breast Cancer (σ=5, threshold=0)


Grayscale


LoG Output

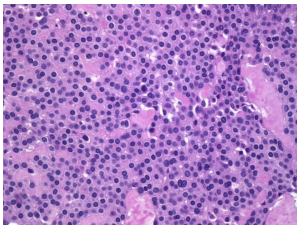

Binary Mask

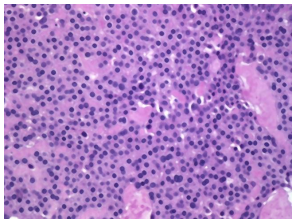
The parameters used in the above experiments seemed to generate the best results from many different iterations with varying parameters. There may have been a better combination out there, but I did not find it and I deem my results satisfactory as is.

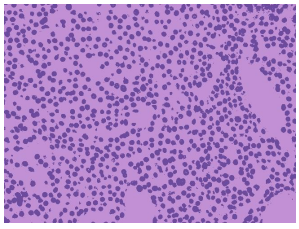
Task 3 - Segmentation

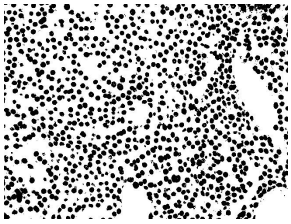
This step was not nearly as hard as the previous due to our ability to use OpenCV's K-means implementation. My feature vector for K-means was both the color and intensity of a given pixel.

Glomus Tumor

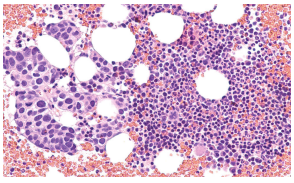

Original

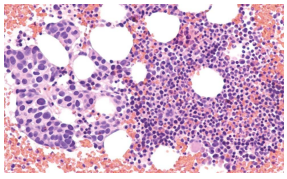

Pre-processed

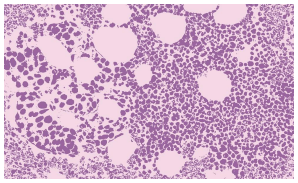

Segmented

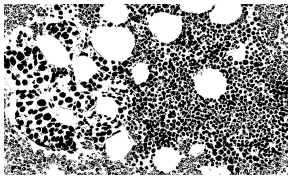

Nuclei Mask

Metastatic Breast Cancer


Original


Pre-processed


Segmented


Nuclei Mask

These outputs are satisfactory as they show the nuclei we are searching for with strong accuracy, likely exceeding the performance of the blob detection. Though, this method (to my knowledge) has the drawback of being significantly slower than the blob detection in the prior task, but I didn't notice any significant slowdowns while working on this.

Conclusions

In conclusion, my implementation of the blob detection and segmentation accurately accomplishes the task of masking the nuclei from the rest of the image. I do think my blob detection is lacking because I didn’t do the automatic scaling for blobs, meaning some blobs aren’t filled in all the way and some are missed altogether. Another improvement that could be made to the blob detection is to do connected component checking, which could fill in all the blobs I missed, but I am not sure how to do this as of yet.