COMPX310-20A Assignment 4

Ye-Gon Ryoo 1126331

Pipeline

The program presented challenges, namely adapting the pipeline to be versatile against variations in image quality. The final version is as follows:

Laplacian difference
→ Median filter
→ Contrast adjustment (fixed 10:150)
→ Median filter
→ Gaussian blur
→ Contrast adjustment (fixed 30:100)
→ Gaussian blur
→ Threshold (150)
→ Erode → Dilate → Dilate → Erode
→ Count regions

First step was to illuminate the image so it can be further processed, and to do so in a way that will produce a consistent output regardless of the input. Two different methods were tested, Laplacian difference, and automatic contrast adjustment using histogram. It came as a bit of a surprise that the former performed better during tests in bringing out the differences between the cells against the background, so I used that option. Median filter treats some of the noise produced as a result of this.

This was followed by a fixed contrast adjustment, which made most of the images much easier to parse through human eyes – certain features of the cell were now highlighted and distinguishable. Any remaining noise that had been amplified by the previous step was cleaned up with a median filter followed by Gaussian blur.

The images are now what I'd consider ideal for visual inspection, but for computational processing, it has to be prepared for thresholding, so the contrast was turned up further. I found that applying thresholding straight away here created a lot of noise around the edges of the cells that presented a challenge, even with multi-depth opening and closing, but an elegant solution was to apply a blur before the threshold.

Opening and closing can be implemented using multiple depths, but this wasn't required, and single-depth of both were implemented to clean up the threshold image.

Counting

While the image processing almost never completely ignores a cell, it is still not quite perfect, and the resulting image has two potential flaws: clumps of multiple cells are counted as a single cell, or a cell has multiple illumination points and registers as more than one white spot. Since I couldn't tune the program to completely solve both of these issues, I tried to balance them – if the occurrences of overcounting was about the equal to the occurrences of undercounting, then the resulting number will be roughly correct. A threshold was added to make sure that only white spots of certain sizes are counted, in case of noise that wasn't eliminated by the opening step.

Performance Accuracy

Image	Actual number of cells	Program estimate	Difference	Difference %
45780	91	87	-4	-4.4%
45799	66	63	-3	-4.5%
46075	111	107	-4	-3.6%
46265	85	91	+6	+7.1%