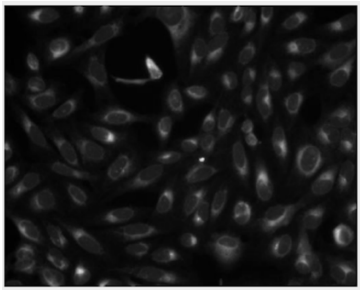
COMPX301 Assignment 4: Pipeline report

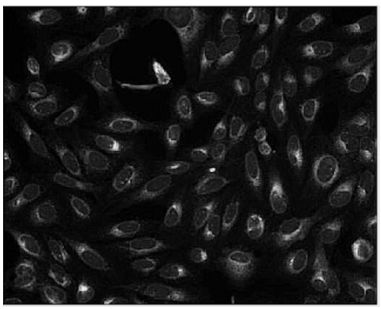
For this report demonstrating the pipeline image processing the following starting image was used:



Part 1: Contrast & Sharpening: In order to start the pipeline we aimed to make the cells more pronounced from the background. We began by adjusting the contrast of the picture to help separate the cells from the background through performing a RescaleOp with an intensity scale of "5" before performing a Sharpening filter on the image through the use of a 3 by 3 kernel (Shown to the Right) with the ConvolveOp operation. We found during our testing that the combination of Contrast and Sharpening worked better to than using different intensity's of Blur and Laplace filter to separate groupings of cells as we faced issues with closely grouped cells being detected as a single cell and when using these methods to process the image:

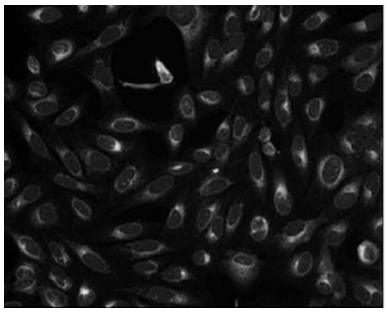
```
float[] sharp = {
        -1, -1, -1,
        -1,  9, -1,
        -1, -1, -1
};
```

After performing these operations on the Initial image the result was as bellow:



Part 2: Average: In order to reduce noise from the sharpened image, we created a method to Average each of the pixels by working out an array of greyscale pixel intensity values of a passed in image and for each pixel x and y coordinate used it find the 9 intensity values (in all 8 directions from a given pixel index including the centre pixel) where all the pixel intensity values were added together with one another and divided by the number of pixels being checked. We initially implemented a Median and Weighted Median method but found the Averaging method to provide cleaner and smoother results resulting in better detection of cells during the regioning stage.

After performing these operations on the Part 1 Image the result was as bellow:

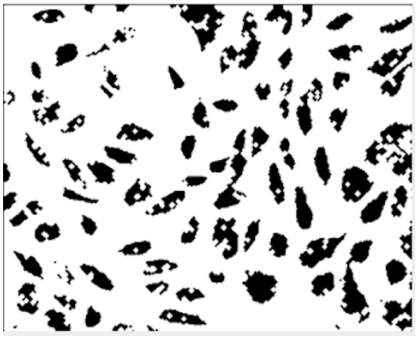


Part 3 Histogram & Threshold: In order to limit the colour space and help in differentiating the cells from the background we created a greyscale histogram of the pipeline image using the scaled intensities (0.2126 for RED, 0.7152 for GREEN and 0.0722 for BLUE to follow with how it is perceived) and 256 different bins. This histogram was created using an array of integers where the integer value at each index represents the total number of pixels which fall under that bin. The Auto Thresholder (A ImageJ plugin feature) is used to determine the threshold value to separate cells from the background. Intensity values greater than or equal to the threshold intensity value would be recoloured to the lowest intensity value, Black (0,0,0) in order to represent cells. All values lower than the threshold value were recoloured to the highest intensity value, White (255,255,255). These set intensity's made the "Flood Fill" algorithm detect areas easily from one another

After performing these operations on the Part 2 Image the result was as bellow:

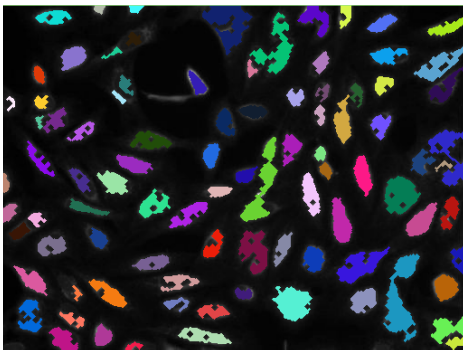
Part 4: Erosion & Dilation: In order to help with separating cells, Erosion and Dilation methods were coded which decrease (Erosion) or increase (Dilation) the size of the detected areas in an attempt to sperate areas of cells which are very close to one another and remove small areas of noise from the Thresholding converting image from 256 intensity values down to 2 intensity values (White and Black). The Erosion and Dilation would affect the +1 -1 Horizontal or +1 -1 Vertical pixel values if beside an opposing colour for each pixel coordinate. During this stage we ran Erosion three times in a row and Dilation once after. Erosion was run 3 times as during our testing this managed to eliminate most of the noise (tiny groupings of pixels around the cells). We did not run it more than 3 times as this would get rid of too much of the detected cells. Dilation was run once to make up a bit for the areas hollowed out of the cells from the white pixel noise in the middle of valid cells, We did not run Dilation more than once in order to prevent close together cells from being re-joined together.

After performing these operations on the Part 3 Image the result was as bellow:



Part 5: Flood Fill & Region Labelling: Once the image had been processed to use Two different colours (Black 0,0,0 representing the Cells and White 255,255,255 representing the Background) and most noise had been removed by Erosion & Dilation. We then ran a Region Labelling algorithm which utilised Recursive Flood Fill Method to find an entire area to label while also recording the PCount (Pixel Count). In order to remove any remaining noise that was picked up by the Flood Fill method we used the recorded PCount Value to determine if any areas were below a set value (50 pixels is the lowest size we deemed to be acceptable given our testing) After eliminating these Areas we would give each area a random colour to help identify separate cells on the output image.

After performing these operations on the Part 4 Image the result was as bellow:



Through our testing we found this pipeline to deliver the most consistent and accurate results across a range of different cell images. While there were a few visual artefacts in the output with missing areas in the middle of cells due to the erosion process, we found this was a better outcome than having inaccurate results and too many cells being merged.