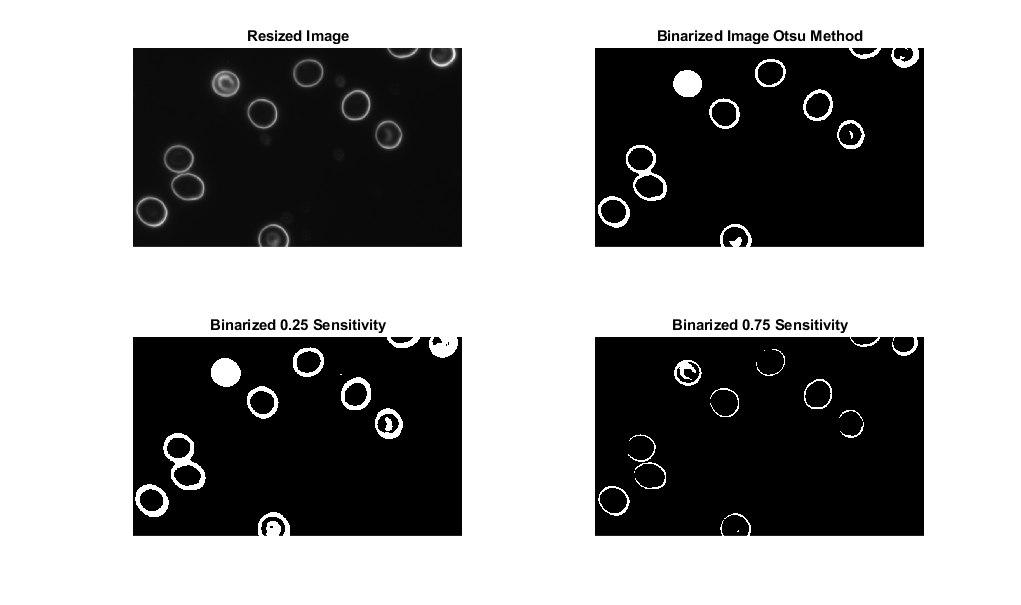
Image Processing Assignment

Youssef Zairi 28475602

# Task 1: Pre-processing:

* + - IMG\_01.png was loaded into MATLAB environment using imread and visualized to observe initial characteristics.
    - The image was then converted to grayscale using rgb2gray, this simplified the analysis by reducing colour complexity whilst retaining the intensity information.
    - The grayscale image was rescaled to a fixed height of 512 pixels while maintaining the aspect ratio using the code shown in the image. This ensures consistent sizes across different images that could initially be different sizes for further processing.
    - A histogram was generated for the grayscale image using 64 bins to show the distribution of intensity values in the image. This histogram was then compared to one created after enhancing the image after enhancing the image using imadjust to observe the effect this had on the intensity values of the image.



* As can be seen by the comparison made of the two histograms Pre and Post Enhancement. The image shows a more spread out distribution of pixel intensity values after enhancement. As a result of this more stretched intensity the image is more contrasted after enhancement.
* After confirming enhancement improved the image’s contrast, the image was binarized using Otsu’s method. To experiment further I tried binarizing the image using manual thresholding. The results can be seen for Otsu, 0.25 and 0.75, after reviewing these Otsu’s Method proved to produce the best binarization detecting the cells clearly and picking up the borders of cells with different brightness. It also picked up the least noise compared to the others.

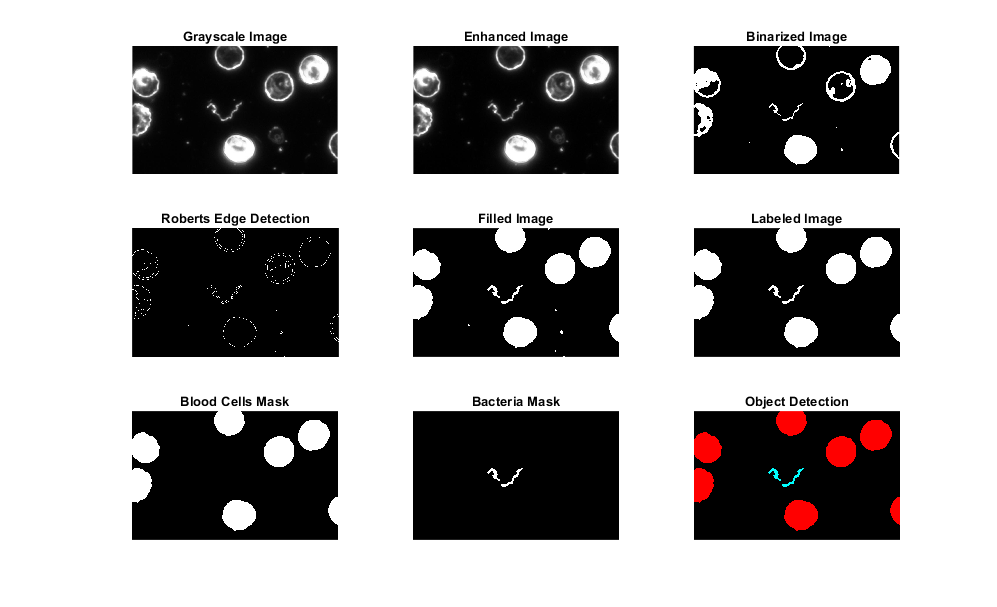
# Task 2: Edge Detection

* To experiment and ensure picking the best edge detection method I compared all four methods, Sobel, Prewitt, Canny and Roberts. After analysing the images they seemed to be similar and so I conducted some research on what Edge Detection method would be best to use in this case. Roberts Edge detection is quick to compute and takes up a small kernel size, guaranteeing fast results and computation which is important in medical imaging.
* Furthermore when testing the edge detection on some of the other images, Roberts Edge Detection proved to be more robust in the context of picking identifying the borders and edges of Red Blood Cells and Bacteria.

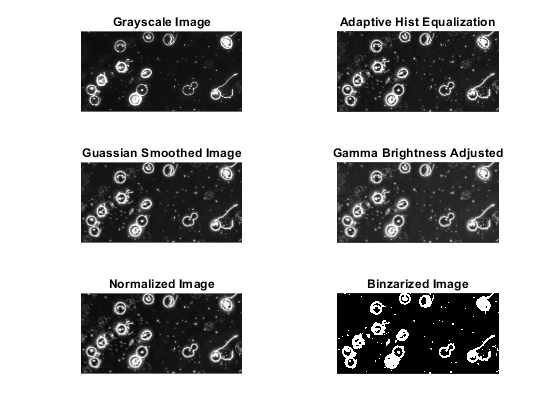
# Task 3: Simple Segmentation

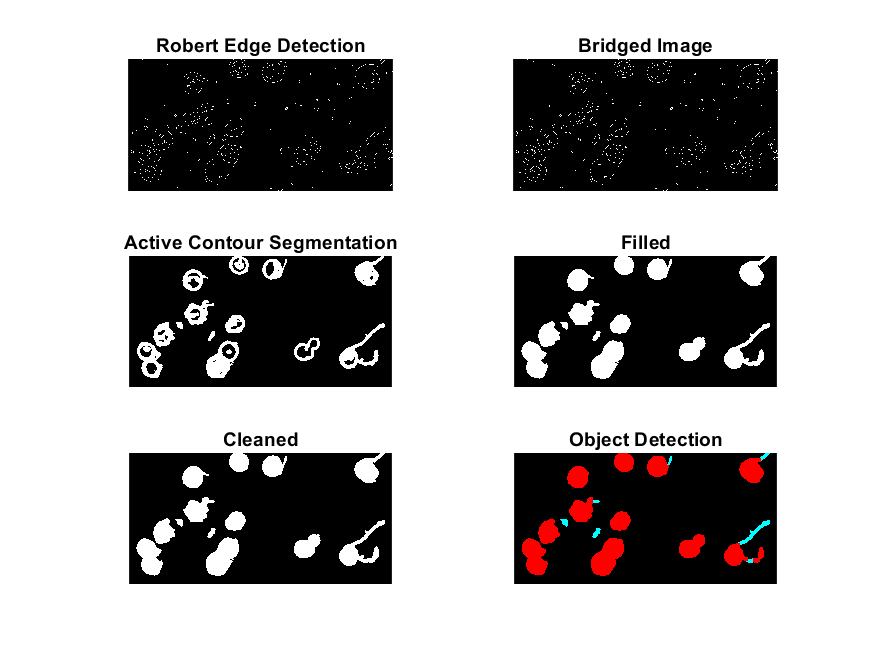
* To Segment the image, first the edged image was filled using the imfill function. When trying to regular function a problem was encountered where the cells in the corner were not being filled properly. To overcome this padding was added to all the corners of the image in order to fix the fill. (Eddins S. , 2013)
* After Filling the image was cleaned in order to remove noise below 50 pixel size and Smoothed.
* The image smoothed image was now ready to be labeled, labeling the image detects the objects in the image using bwlabel ,detecting objects was important as regionprops was used on them to detect charectaristics such as Area to determine whether the object was a blood cell or noise and bacteria.
* The threshold > 2000 Area was concluded after reviewing the area of the objects detected in this image.
* Through this the blood cell mask and bacteria masks were created, combining them to make the Final Segmented Image as can be seen.

# Task 4: Object Recognition

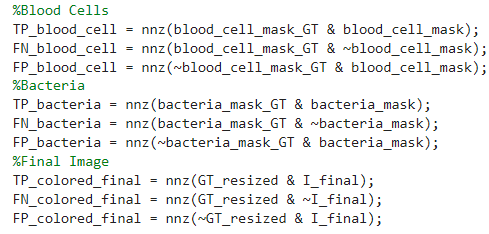
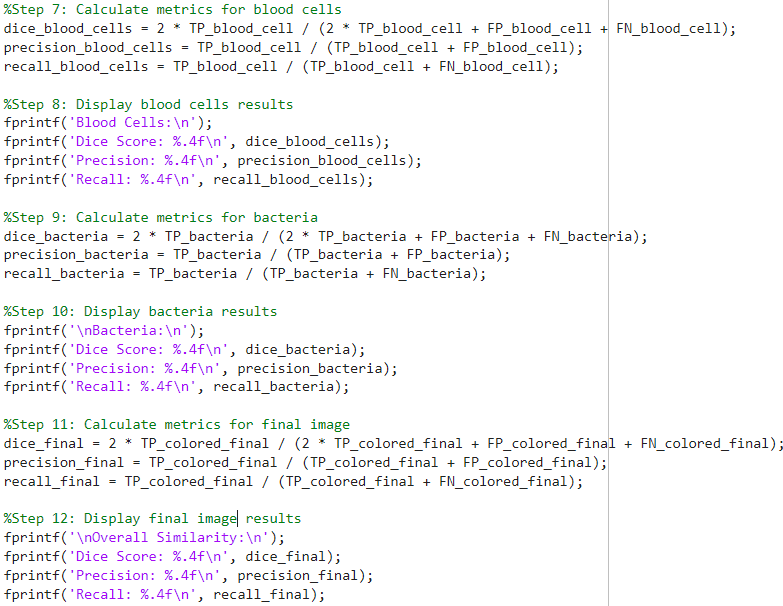
* The steps taken from Tasks 1 – 3 were repeated on the IMG\_11 in order to arrive at the labelled and segmented image.
* This image contains bacteria and so I had to expand the stats being extracted by region props as can be seen in the screenshot above.
* After observing the stats above and testing different parameters to determine whether an object is a Blood Cell or Bacteria, I concluded using the Area in combination with the circularity of the object, calculated by (4 \* pi \* areas) ./ (perimeters .^ 2), was most suitable for correct object recognition. (MathWorks, 2025)
* The blood\_cell\_mask created was objects in the labelled image with an Area > 2000 and Circularity > 0.50. The bacteria mask was the inverse of this.
* This proved to work really well as it detected the Blood Cells and Bacteria perfectly. The next step was to create an RGB image with a black background, colouring the blood cells red and the bacteria with green + blue to create cyan as requested.

# Task 5: Robust Method



* When creating a robust method that works for the majority of images provided in the dataset. I quickly realized that pre-processing and binarization was one of the most important methods, as determining a method that would work for images of different Intensity values was going to be the main hurdle.
* After testing many different binarization methods such as adaptive thresholding with different sensitivity and manual thresholding, I realized Otsu’s Method was best to use but the image had to be enhanced thoroughly before binarization.
* After Resizing the image and converting it to grayscale, I applied Adaptive Histogram Equalization using adapthisteq. There was lots of testing involved with this to find the correct Number of Tiles and Clip Limit parameters. I concluded that NumTiles [32 32] and ClipLimit 0.01 was best in order to maximise the equalization in blocks of 32x32.
* Next, I used Gaussian filter with a sigma level of 2 with the command imgaussfilt and smoothed the image using Gamma of 0.8 in order to brighten the image more.
* Finally, I normalized the image using mat2gray function before binarizing it with Otsu’s Method. Otsu’s method after the pre-processing mentioned above to adjust the intensity and brightness of the images proved to be a robust binarization method.
* After Binarization the same Roberts edge detection was applied to the image.
* To achieve a more robust method in segmentation, the edged image had to undergo some morphological processing, I used imclose function to close small gaps defined by an se\_close scale of strel(‘disk’,3). This defined the size of gaps that were closed. The image was then Bridged using the bridge function under bwmorph.
* For segmentation, the previously used method was not proving to be robust and there were lots of issues run into when segmenting. After some research I found that active contour segmentation can be more robust and it proved to be. First, we create an initial mask to define the region of interest in the image, the activecontour model is then used to refine segmentation by evolving a contour to fit the boundaries of objects detected in the image. Finally, bwareaopen is then used to remove noise smaller than 700.
* After having our segmented image, filling the image and ensuring objects on the borders are filled is next step, instead of the previously used method, I decided to only fill top\_bottom padding and left\_right padding and not all corners to avoid issues that came up with image 08 having bacteria in the corner.
* For Object Detection to be more robust I decided to change the method used, instead of using regionprops for stats on objects. I decided to use bwareaopen(imopen(logical(the cleaned segmented image), defining a strel(‘disk’, 14) and a size requirement of 2500. This took objects with radius 14 and size 2500+ as blood cell mask. All other objects above 700 pixels were the bacteria mask. After joining these 2 layers and creating an RGB image the same method as Task 4, I had a functioning Robust Method.

# Task 6: Performance Evaluation

* In order to evaluate my object detection against the ground truth accurately, the GT image was resized to match my final image using imresize(GT,[size(I\_final, 1), size(I\_final,2)]); This resized both layer 1 and 2 of the GT image to match the size of my final image.
* In order to evaluate my processing more in depth, I decided to evaluate the blood cell mask, the bacteria mask as well as the over all image. To accomplish this, I converted the GT Images to a blood\_cell\_mask\_GT and a bacteria\_mask\_GT taking the respective layers from the resized GT image.
* The metrics chosen to evaluate my model’s performance were Dice Score, Precision and Recall. In order to calculate these, I would need the True Positive, False Negatives and the False Positives of my predictions or detections against the ground truth. To calculate these, I used nnz which is a matlabs function that returns the number of non-zero elements from a matrix. (Mathworks, 2025) Using this in combination with bitwise operations made calculating these easy.
* After calculating these I calculated the chosen metrics, Dice Score, Precision and Recall for each of Blood Cell Layer, Bacteria Layer as well as the final image. I applied this evaluation to all of the images in the dataset from IMG01 to IMG15 and recorded my results. To calculate and display each metric, the following formula’s and code was used.
* This method proved to be efficient and provide an easily interpretable result that was recorded for each image on an excel sheet and output onto a table shown below.
* The Final Image overall evaluation shows that the model’s average **Dice Score** **= 0.8219 Precision = 0.7186067 Recall = 0.97746**.
* Considering this model solely relied on Morphological and non-machine learning functions to create a robust image processing object detection method, this was is a successful model as proven by the metrics provided.

# References

Eddins, S. (2013) *Defining and filling holes on the border of an image*, *Steve on Image Processing with MATLAB*. Available at: https://blogs.mathworks.com/steve/2013/09/05/defining-and-filling-holes-on-the-border-of-an-image/ (Accessed: 15 January 2025).

*Label components using 4-connected objects* (no date) *MathWorks*. Available at: https://uk.mathworks.com/help/images/ref/bwlabel.html (Accessed: 15 January 2025).

*Remove objects in image containing fewer than 50 pixels* (no date) *MathWorks*. Available at: https://uk.mathworks.com/help/images/ref/bwareaopen.html (Accessed: 15 January 2025).

*Roberts cross edge detector* (no date) *Feature Detectors - Roberts Cross Edge Detector*. Available at: https://homepages.inf.ed.ac.uk/rbf/HIPR2/roberts.htm (Accessed: 15 January 2025).

*Calculate centroids and superimpose locations on image* (no date) *MathWorks*. Available at: https://uk.mathworks.com/help/images/ref/regionprops.html (Accessed: 15 January 2025).