Detecting Cell Nuclei

StudentID : 20186501

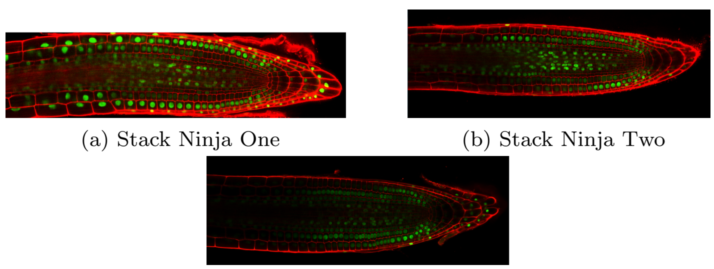
17 March 2022

1 Introduction

Within the field of biological sciences, there is various work done involving investigative measures on microscopic images. These images, will often need to be further analysed through Image processing. We can find that the Matlab Image Processing Toolbox provides necessary facilities to solve real world problems. In this report we will be using Matlab to change image data,

2 Strategy

The images that will be investigated are of plant roots (Figure 1.) which is taken through a confocal laser microscope. In the following manner, we will discuss strategies which, samples the image colour space, reduces image noise, applies thresholding conditions and processes the image into a binary image product. The overall goal being to explore a possible software solution to extract regions of nuclei as a binary image.



(a) Stack Ninja One

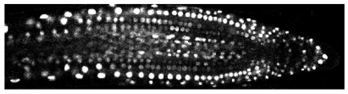
(b) Stack Ninja Two

(c) Stack Ninja Three

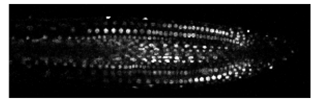
Figure 1: Test Sample

2.1 Colour Space Conversion

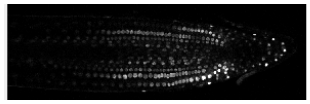
With a first examination of the image samples in figure 1. we can clearly see that it is a coloured image. Coloured images have three values stored in a 3D array, which is a model based on the RGB standard. As such images can be separated into three components red, blue and green where the arrays can be accessed to apply changes. For the purpose of this investigation we can ignore the colours completely, and just extract data related to the nuclei. Looking into the samples all the nuclei emits a bright green colour. Therefore the single array from the green channel can be pulled out, which is easier to process than a whole 3D array. Giving us a grey scale image as a base to allow further image processing techniques.



(a) Stack Ninja One



(b) Stack Ninja Two



(c) Stack Ninja Three

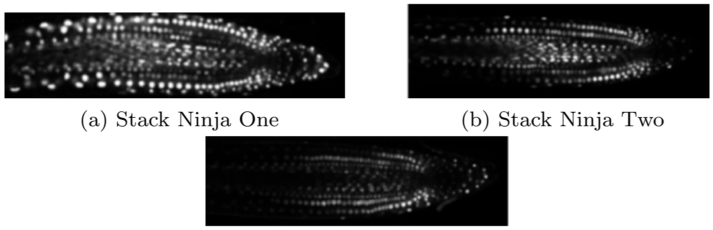
Figure 2: Grey level Image

2.2 Noise Reduction

Although the picture was taken through a very sophisticated scope it is all the wiser to consider all aspects of the image. As images are never perfect, there will always be errors in image values. These errors are known as image noise, which is collected through imperfect sensors when taking the picture. There are also other reasons such as compression which would introduce noise during the process. Looking closely at figure 2, it can be addressed that the quality of the nuclei is not the clearest and there may be fragments of noise which can be removed. There are noise reduction strategies which can be used to tackle this issue.

A common method would be through convolution which takes the average value over a spatial filter. Mean filtering would be a good example of filtering through convolution, although simple it has a huge drawback. Noise pixels range differently from the targeted pixel value, thinking about it in that manner the average pixel calculated would reduce sharp edges. Adding to it even though the noise is removed, it leaves us with a blurry image which is not ideal. Mainly from image values that is not close to the true value being considered onto the calculation.

To build on this, a Gaussian filter is developed which reduces these uncertainties. The Gaussian filter is similar to mean filter, however the array of weights is determined through the Gaussian distribution. In short, the closer to the source pixel the value is the more reliable it becomes. Filtering works on the basis of the sigma value. This method is one which can be considered to apply onto the grey level images. Through Matlab adjusting the value of the sigma is key. The test conducted was to use sigma values from (0.5, 1, 1.5 ... 4.0). Where it was concluded that values closer to 0 didn't seem to have a significant visual change. Whereas, values closer to 4.0 lost a lot of it's edges, giving a more blurred representation (Figure 3.). However this requires a more manual approach with trail and error to figure out how much filtering needs to be applied.

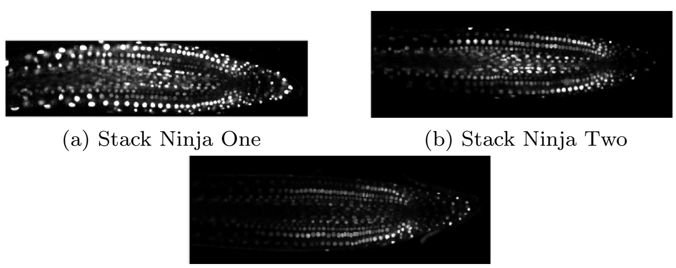


(a) Stack Ninja One (b) Stack Ninja Two

(c) Stack Ninja Three

Figure 3: Gaussian Filtering

Another approach to reduce the noise would be through median filter, overall this approach takes less time and effort over. The median filter (Figure 4.) is a nonlinear filtering strategy, unlike its counterpart the mean filter which uses linear filtering techniques. The aim of using this filter is to preserve edges and reduce as much noise as we possible can. The process can be expected to take the median pixel value from sorted pixel lists involving neighbouring pixel values. In theory, this will ensure abnormal pixel values are not selected during the filtering process giving a more realistic pixel data. A main advantage of the median filter is to reduce input noises in a larger magnitude.However, there are also disadvantages that succumb to this approach. There is no proper error propagation therefore analysis of the method tends to be difficult and it can also be noted that this feature isn't perfect and should be used to appropriate noises.



(a) Stack Ninja One (b) Stack Ninja Two

(c) Stack Ninja Three

Figure 4: Median Filtering

Through the evaluation of these strategies, the median filtering approach gives more reliability and preserved more of the image details. Moreover, it requires less work and the process in general is more automated giving us a good base to work with.

2.3 Threshold

Now that there is a base, we need an easier application of the image for a more clear analysis. We can judge the base to hold light objects (nuclei) and a black background. A process called thresholding helps us with this where image pixel can be manipulated in order to target corresponding objects. In practise, pixel intensity values are taken and values which are above or below the threshold are placed into sets of foreground and background classification. This can be adjusted manually to test a hypothetical threshold point, where if the threshold is deemed too low, pixels will be considered background and if it's too high pixels will be considered foreground. This makes the adjustments of the threshold value quite sensitive.

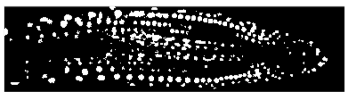
Hence, this process can be looked at to be too simple and mistakes can be easily made. Also note that manual threshold assumes no overlap between two regions brightness which is not realistic. Concluding this strategy to be not applicable in this analysis. Overall it is a very tedious operation which relies heavily on trail and error which doesn't helps it cause.

On the other hand, a better way of working can be used which is through adaptive threshold which gives a more consistent result. This is where measurements is taken through histograms of the image and the threshold can be altered dynamically making a close assumption to the ideal threshold. The method I opted for is to use the Otsu thresholding. This model assumes the histograms are bimodal which has two regions for background and object and in our case we have a base image with a black background and a white object. This is an automated process, where threshold values are computed, calculating a spread of pixel levels on either sides of the threshold. Where anything above and below a certain threshold will be cleared to be a pixel value that is foreground or background. The goal here is to find a threshold which gives a minimum weighted sum between the two foreground and background region.

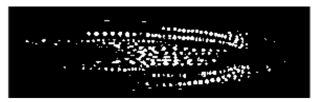
Another thing to mention would be unimodal thresholding, through a quick judgement the assumptions made in this model wouldn't be match for the image we have. As characteristics of a unimodal image include most of the base being filled with background but just a tiny portion of object present, and vice versa. These types of images would be more beneficial using a Rosin's algorithm which is contrasting to Otsu's algorithm as they are defined for different approaches. From this we can further conclude the selection of Otsu to be the correct one.

2.4 Binary Image Processing

Finally the image can be binarised using the threshold value that has been retrieved through the previous process. This is the end goal of the assessment where we want the most optimal binary image. We can explore two approaches regarding the the adaptive threshold. First approach is to use the threshold globally. In this instance (Figure 6) we can see that althought the results seems to pick out the brighter objects, the faint ones are not captured at all. This may be a result of a strong illumination gradient specifically in Stack Ninja Two and Three. As such parts of the images wont apply Otsu's.



(a) Stack Ninja One



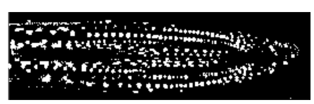
(b) Stack Ninja Two

(c) Stack Ninja Three

Figure 5: Global Adaptive Threshold

A solution that can be derived from this is to use a local adaptive threshold approach (Figure6.). This is the practise to break down images based on the illumination across the image as sensitivity differs going across. Therefore we can apply Otsu individually to smaller broken down regions of the image then finally merge back to a whole picture.

(a) Stack Ninja One



(b) Stack Ninja Two

(c) Stack Ninja Three

Figure 6: Local Adaptive Threshold

Through comparison between local and global adaptive threshold. It can be clearly seen that there are significant differences. Mainly being the that the local adaptive extracts more objects successfully. This is visibly very clear on Stack Ninja Two and Three, again this can be pointed back to the illumination gradient as the main factor. However, the local adaptive images, produces faint image noise which looks like salt and pepper noise. This may be due to the steps being taken throughout the investigation. For example a use of a higher contrast during noise reduction could've affected the product or maybe just applying a different strategy can be used to fix the problem.

There are different strategies which can be used to get to close to the ideal binary image product. These methods needs to be thoroughly analysed and considered, whilst evaluating the advantages and disadvantages. Also we need to understand that it is near impossible to get the 'perfect image', there will always be a form of collateral damage when processing images. Therefore with each image appropriate strategies need to be discussed and look at which drawbacks are limited enough for us to be satisfied with the results.