Extracting & Analysing Cell Nuclei

24th April 2021

Name : Liew Yih Seng

Student ID : 20090325

Email : hfyyl2@nottingham.edu.my

Module Name : Introduction to Image Processing (COMP 2032)

Module Convenor : Dr. Amr Ahmed

Link to Demonstration Video : https://youtu.be/Vezm12tVfjA

Table of Contents

Introduction

Confocal Laser Scanning Microscopy (CLSM) is a technique of microscopy imaging that has been heavily used in the field of biology. These biological specimens are required to undergo a fluorescent dye, illuminating the targeted components existed within the specimen under laser light, providing researchers with a clearer view to the structure of their specimens. This paper proposes a pipeline that performs extraction and analysation of cell nuclei from CLSM images through utilizing various image processing operation available in Matlab.

Methodology

As proposed in the coursework sheet, the aim to extract and analyse cell nuclei can be achieved by undertaking four main processes within the proposed pipeline: (a) Colour Space Conversion, (b) Noise Reduction, (c) Thresholding/Segmentation and (d) Binary Image Processing. Below comprised the discussion of each step within the pipeline.

(a) Colour Space Conversion

The first step is to select a suitable colour space to work on. During initial approaches, extraction of green channel from RGB colour space has been used as it is straightforward that all targeted components to be extracted are of green colour. However, repetitive explorations into other colour spaces have proven that working with RGB is not the optimal solution. The extraction of hue channel from HSV colour space has resulted in a clearer output when compared to the extraction of green channel from RGB.

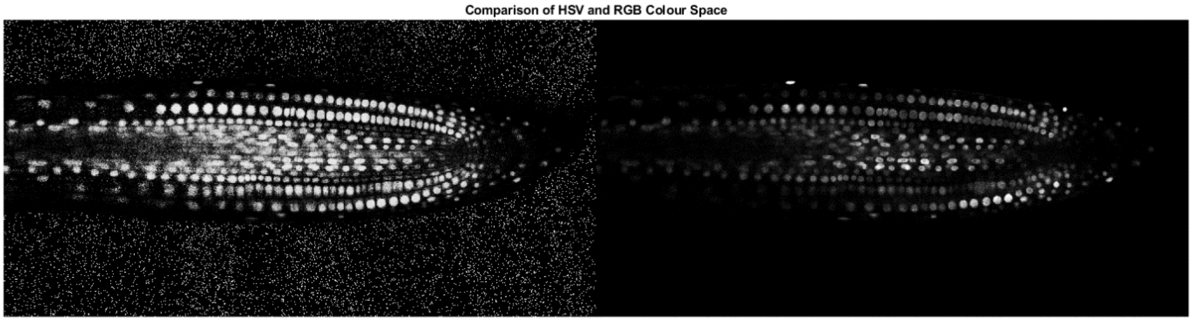


Figure 1 shows comparison of colour space conversion obtained from hue channel of HSV (left) and green channel of RGB (right)

Referring to Fig. 1, it can be observed that (Fig. 1 left) illustrates an evenly amount of contrast over the entire image. Despite having more noise, (Fig. 1 left) demonstrates better representation of the detected cell nuclei especially on the left and right regions of the image when compared to (Fig. 1 right). Besides, (Fig. 1 left) offers a better clarity to all the detected nuclei. Meaning, the detected nuclei in (Fig. 1 left) are of the same proximity of white intensities when compared to nuclei detected in (Fig. 1 right) where some nuclei were too dark to be observed.

This can be elucidated by the concept of HSV where it allows the decoupling of value channel, representing the brightness of colour, away from colour-carrying information channel (hue and saturation) in the original image. For the case of RGB, images are composed by stacking three distinct monochromatic images from different channels (red, green and blue), resulting in a limited representation of colour description. Within this paper, hue channel from HSV has been selected because it exclusively contains all the attribute values representing the pure colour that exists within the colour wheel. However, these images provided consisted mainly of red and green values, but illustration in (Fig. 1 left) appeared to have only display the green values existed within the image in grayscale. This is because value that corresponds to the red's position on the colour wheel is very low. In Matlab's representation of HSV, the hue channel describes a colour based on a scale of double, ranging from 0 to 1. Similar to the 3-D cone representation of HSV, the double value in Matlab can be translated into the angle of rotation within the HSV cone, where the colour gradually changes from red to orange, yellow, green, cyan, blue, magenta and back to red as the double value increases. Meaning, there still exists element of red colour from the cell wall inside (Fig. 1 left). The reason for them to not being displayed is because their values are certainly too small causing them to be insignificant during this stage of image enhancement. Therefore, this paper suggests the usage of HSV colour space in the colour conversion method.

(b) Noise Reduction

As shown in Fig. 1, image (Fig. 1 left) is a relatively noisy image especially at the surroundings and centre part of the plant root. Several noise reduction methods such as Median Filtering, Gaussian Filtering, Average Filtering and Anisotropic Filtering have been implemented.

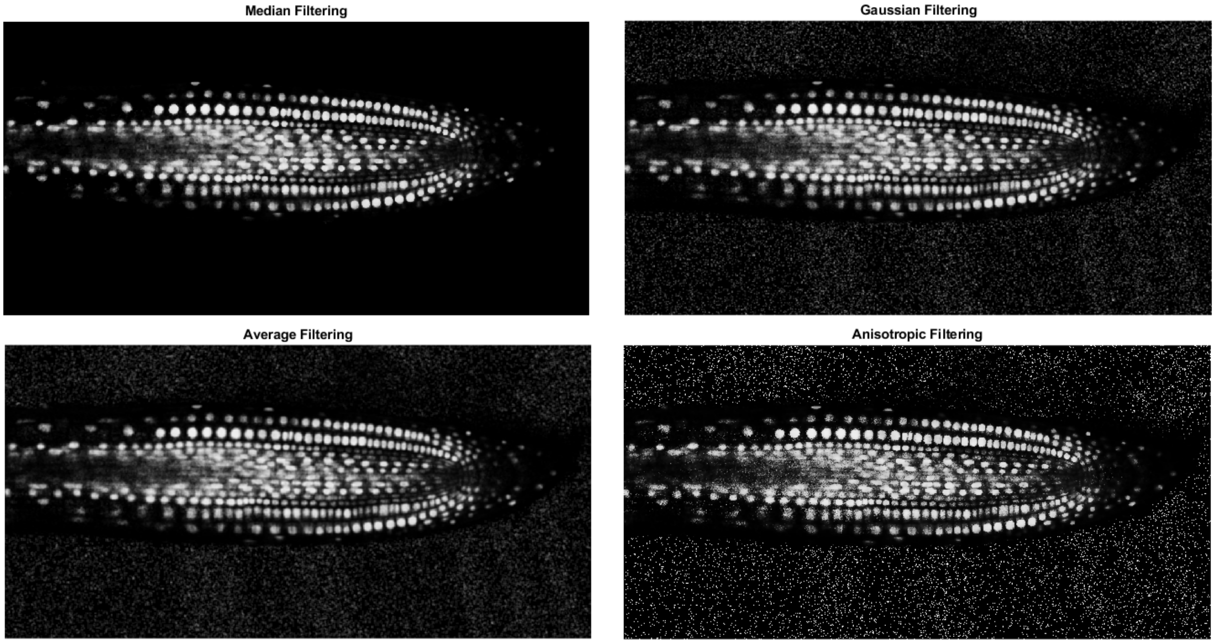


Figure 2 illustrates comparison of four kinds of noise reduction method being applied onto the image. (Top-left: Median Filtering with neighborhood 4x4, Top-right: Gaussian filtering with sigma 0.8, Bottom-left: Average Filtering with filter size 4x4, Bottom-right: Anisotropic Filtering)

From the comparison above, it is obvious that (Fig. 2 Top-left) has the best outcome among all four noise reduction methods. Median Filtering has outperformed other noise reduction methods and has proven to have effectively reduced most 'salt' noise present at the surroundings of the plant root. Median Filtering is classified as a non-linear filter that performs the process of smoothing without causing excessive blurring onto the original image when compared to linear filters. By comparing all centre regions of plant roots in (Fig. 2), it is noticeable that most nuclei in (Fig. 2 Top-left) still retain their original shapes even after the process of smoothing to eliminate noise present within the centre region of (Fig. 1 Left). Therefore, it is undeniable that Median Filtering is the best choice being applied onto (Fig.1 Left).

(c) Thresholding/Segmentation

Implementation of Median Filtering has resulted in an image that visually contains less amount of noise. The term visually has been used because there still exist traces of red pixels within the image after being applied with Median Filtering. To verify, an experiment has been carried out where all images provided has been binarized with threshold obtained from a Matlab function of adaptive thresholding, adaptthresh().

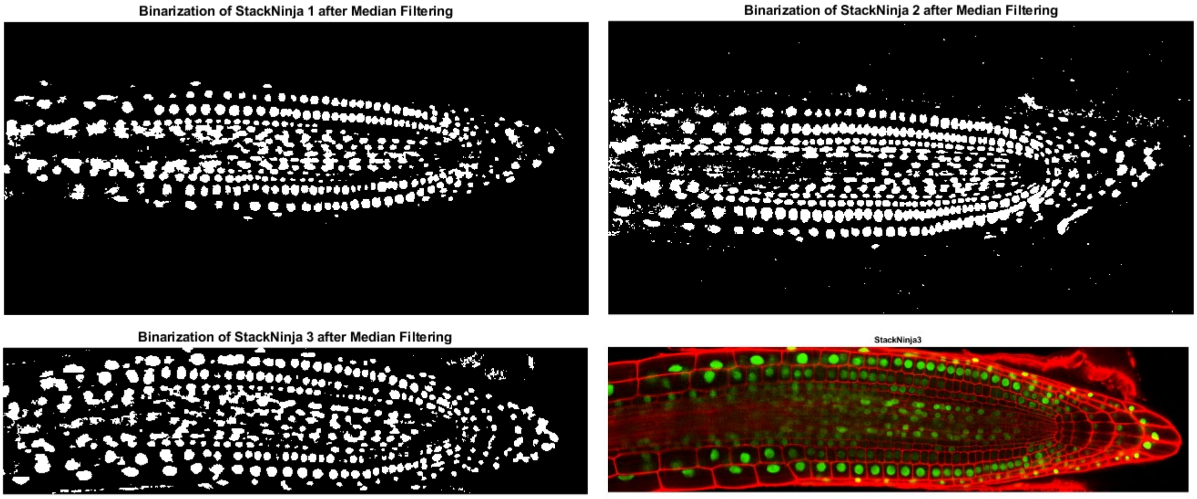

Binarization of StackNinja 1 after Median Filtering
Binarization of StackNinja 2 after Median Filtering
Binarization of StackNinja 3 after Median Filtering
StackNinja3

Figure 3 illustrates comparison of binarization of all images provided that has undergone Median Filtering. (Top-left: StackNinja1.bmp, Top-right: StackNinja2.bmp, Bottom-left: StackNinja3.bmp, Bottom-right: original image of StackNinja3.bmp)

With reference to (Fig. 3), it is evident that traces of red cell wall is present in the upper-right region of (Fig. 3 Bottom-left) when compared with (Fig. 3 Bottom-right). Considering that the range of double value representing red colour within hue channel is known to be low, these red pixels can be removed by executing a for-loop that runs through the matrix of (Fig. 2 Top-left). In this paper, the approach of modifying all double values less than 0.1 to 0 in (Fig. 2 Top-left) has led to an image with lesser trace of red elements.

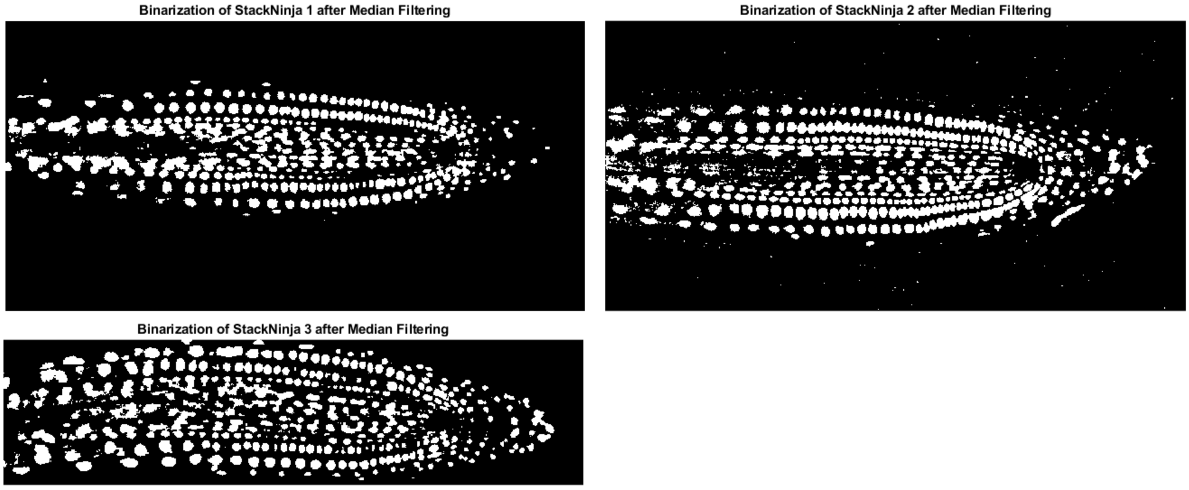

Binarization of StackNinja 1 after Median Filtering
Binarization of StackNinja 2 after Median Filtering
Binarization of StackNinja 3 after Median Filtering

Figure 4 illustrates comparison of binarization of all images provided that has undergone median filtering followed by a simple red pixels filtering. (Top-left: StackNinja1.bmp, Top-right: StackNinja2.bmp, Bottom-left: StackNinja3.bmp)

Shown in (Fig. 4), with simple for-loop filtering modifying values lesser than 0.1 to 0 from matrix (Fig. 2 Top-left), it has been proven that most red pixels values existed within the image has been eliminated, leaving mostly green pixels values inside the matrix.

To discuss how red pixels affect the determining of suitable threshhold value, the mechanism of adaptthresh() has to be known. It computes a local adaptive threshold value for every pixel within the image by evaluating a local mean intensity for neighbourhood of every pixel. With presence of red pixels value in (Fig. 2 Top-left), it is clear to state that these red pixels has contributed into the computation of mean value within the neighborhood, resulting the illustration in (Fig. 3 Bottom-left). Furthermore, this function can be enhanced by inputting the sensitivity value, where it helps deciding which pixel’s intensity will be thresholded as foreground and background pixels. Selecting the right sensitivity is important as value that is too low will result in more pixels including the desired foreground pixels being classified as background pixels. In this paper, the sentitivity value has been set to 0.4 as it performs the best in differentiating foreground and background pixels (Fig. 5).

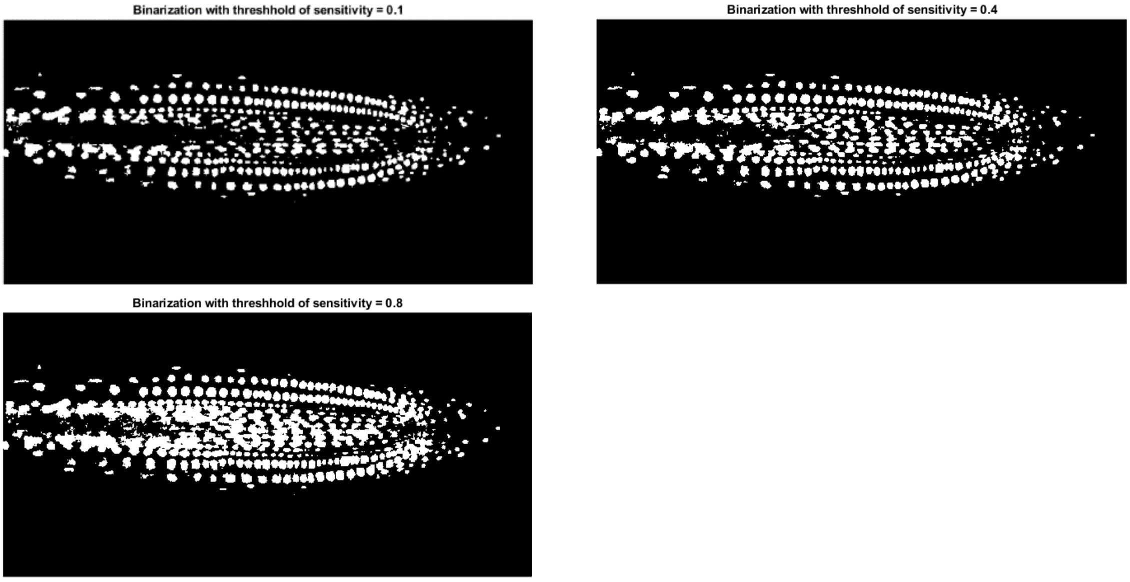

Binarization with threshhold of sensitivity = 0.1
Binarization with threshhold of sensitivity = 0.4
Binarization with threshhold of sensitivity = 0.8

Figure 5 displaying same image undergoing different value of sensitivity in adaptthresh(). (Top-left: image with sensitivity 0.1 loses too much detail in the centre region, Top-right: image with sensitivity 0.4 has an acceptable range of foreground and background pixels, Bottom-left: image with sensitivity 0.8 resulted in lumps of foreground pixels in centre region)

(d) Binary Image Processing

There still exist non-nucleus pixels that are represented in white, especially in (Fig. 4 Top-right) where there exists unwanted pixels in the centre and top-right region of the image. This misclassification of pixels has been treated using imopen() function, consisting morphological operations of eroding followed by dilating using structure element of size 4. This process has significantly removed all small and unwanted pixels (Fig. 6) while retaining the original shape of the detected nuclei (Fig. 7).

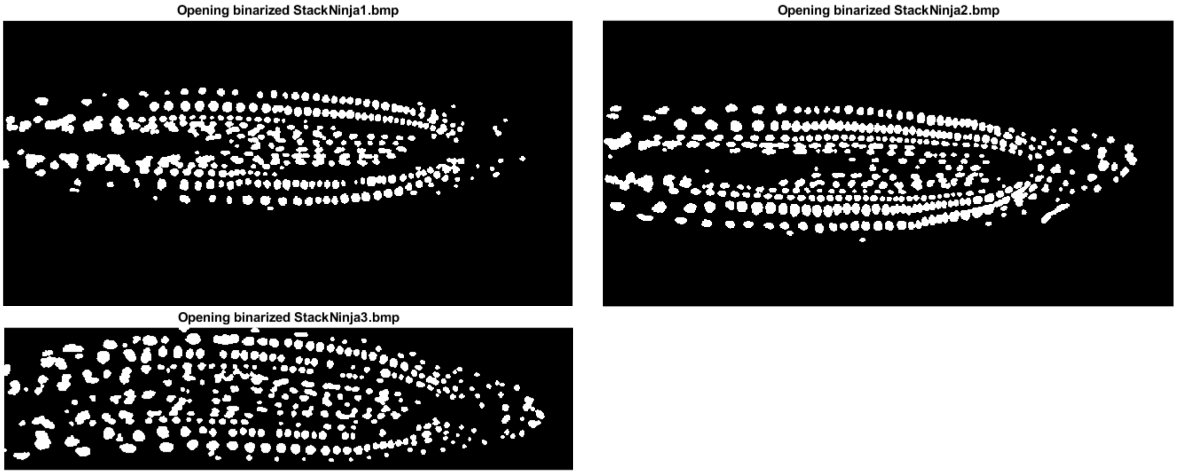

Opening binarized StackNinja1.bmp
Opening binarized StackNinja2.bmp
Opening binarized StackNinja3.bmp

Figure 6 illustrates implementation of imopen() onto all images provided. (Top-left: StackNinja1.bmp, Top-right: StackNinja2.bmp, Bottom-left: StackNinja3.bmp)

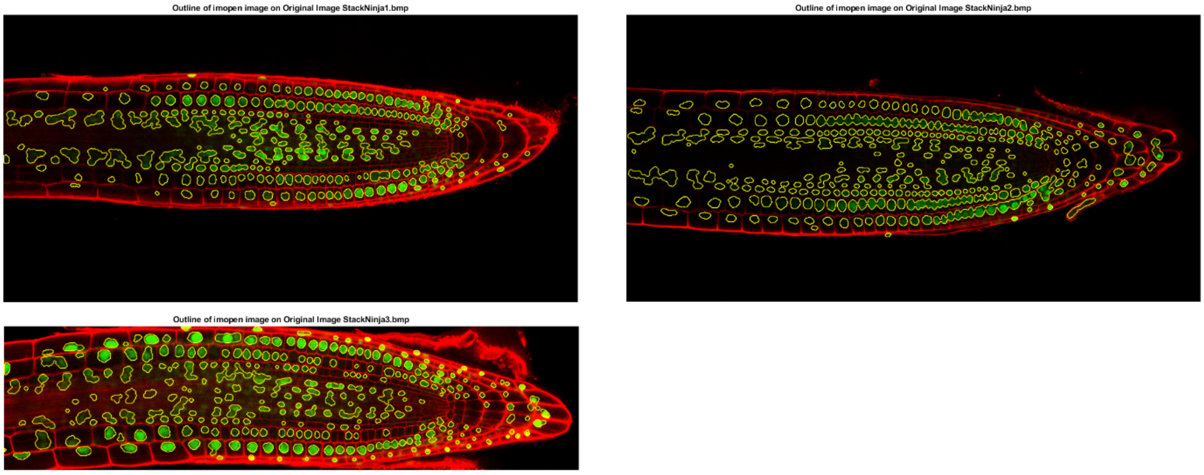

Outline of imopen image on Original Image StackNinja1.bmp
Outline of imopen image on Original Image StackNinja2.bmp
Outline of imopen image on Original Image StackNinja3.bmp

Figure 7 shows outline of imopen in yellow with structural element of size 4 on images provided. (Top-left: StackNinja1.bmp, Top-right: StackNinja2.bmp, Bottom-right: StackNinja3.bmp)

Images in (Fig. 7) shows that most nuclei have been detected, nevertheless there still exist nucleus that has no clear-cut between each other especially in (Fig. 7 Top-left) on the left region of the image where there exist nuclei that are attached to each other. This issue has been solved using watershed transformation where it differentiates catchment basins with ridge lines that aids the algorithm to decide if the detected region is a continuous region or distinct objects (Fig. 8).

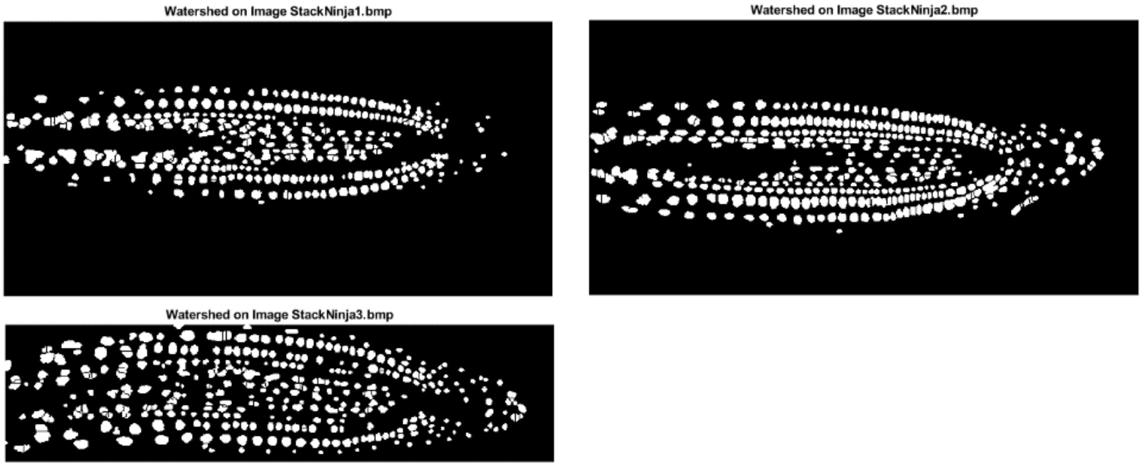
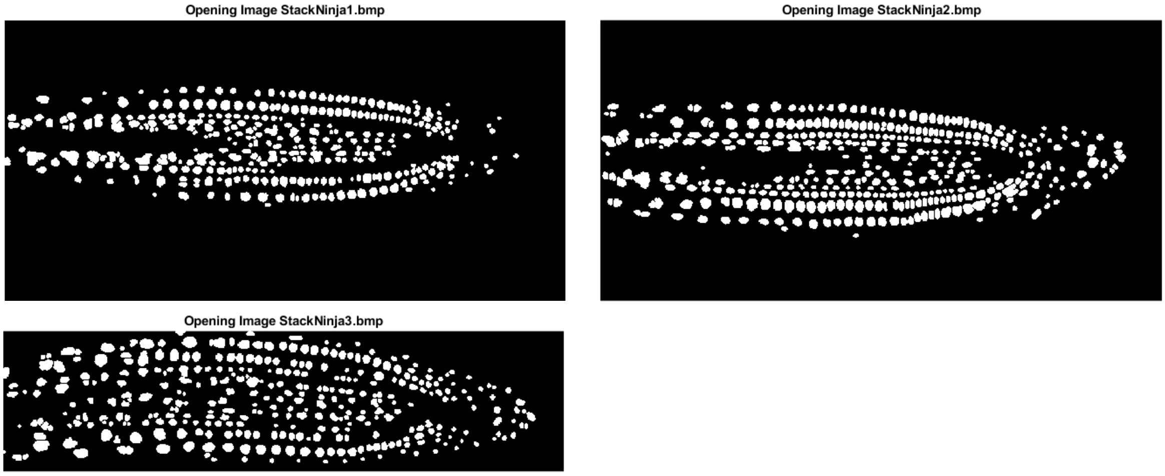

Watershed on Image StackNinja1.bmp
Watershed on Image StackNinja2.bmp
Watershed on Image StackNinja3.bmp

Figure 8 illustrates output of all images that has been applied with watershed transformation. Most continuous region existed within Fig. 6 has been separated into distinct objects

As seen in (Fig. 8) that most continuous nuclei discussed above have been seperated. To enhance the border of all nuclei, the image is again being morphologically opened using structure element of 4.


Opening Image StackNinja1.bmp

Opening Image StackNinja2.bmp

Opening Image StackNinja3.bmp

Figure 9 illustrates final output of extraction of nuclei from all images provided

Analysis

a) Nucleus Count

The count of total nuclei exist within each image has been retrieved using bwconncomp() algorithm that searches for connected components within a binary image.

b) Nucleus Size

The detection of nucleus sizes has been facilitated by function regionprops() with 'Area' as property where given an input binary image (any image from Fig. 9), it returns the actual number of pixels that has been bounded within each separated regions in the form of scalar value.

c) Nucleus Shapes

The analysis of nucleus shape has been accomplished using regionprops() with 'Eccentricity' as property, returning a scalar value between 0 to 1. The function retrieves the foci and the greatest axis length of each detected nucleus, then compute the distance ratio between both values, resulting in a value that determines the likelihood of a nucleus to be circular when result returns 0, or ellipse when result returns 1.

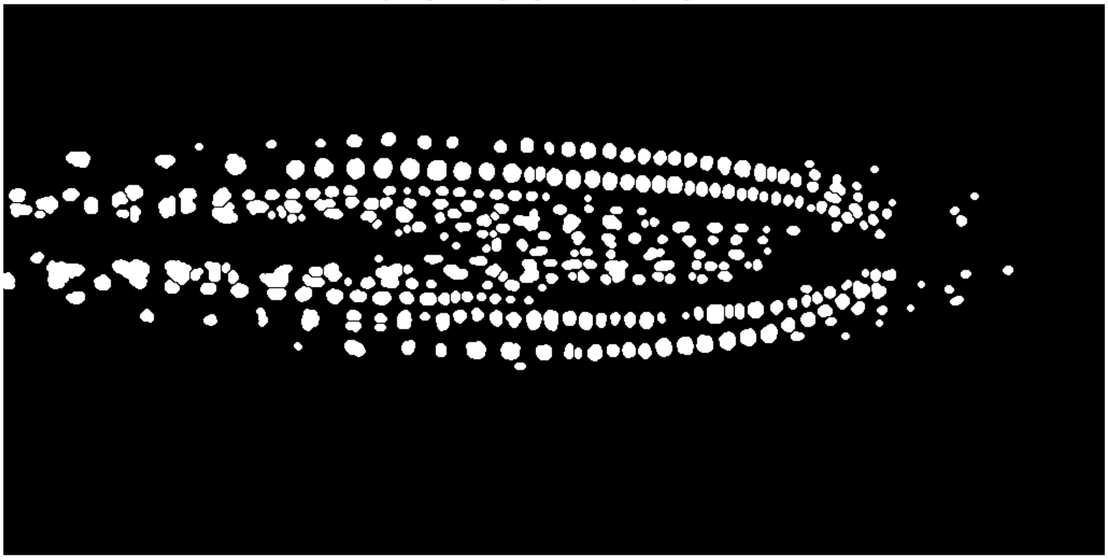
d) Nucleus Average Brightness

Detection of nucleus brightness is done by retrieving all pixel coordinates of detected nuclei using 'PixelList' in regionprop(), followed by extraction of brightness value from the value channel of HSV based on the pixel coordinates obtained from regionprop().These brightness values within each nucleus will be sum then divided by the total number of pixels counts of each nucleus to obtain the average brightness of each detected nucleus.

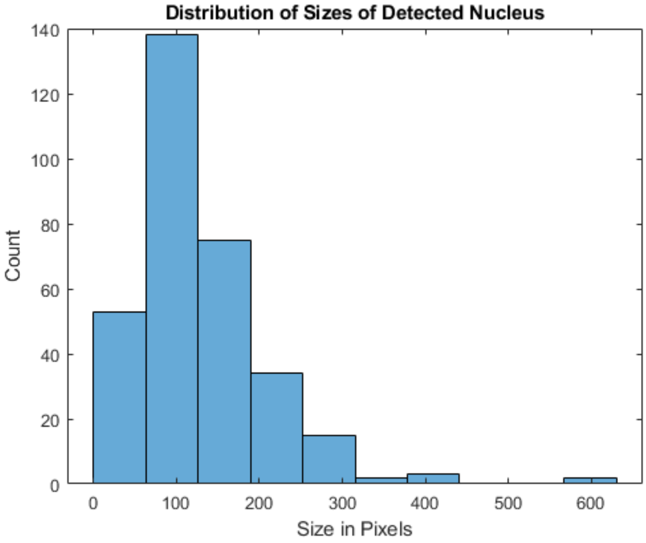
Results

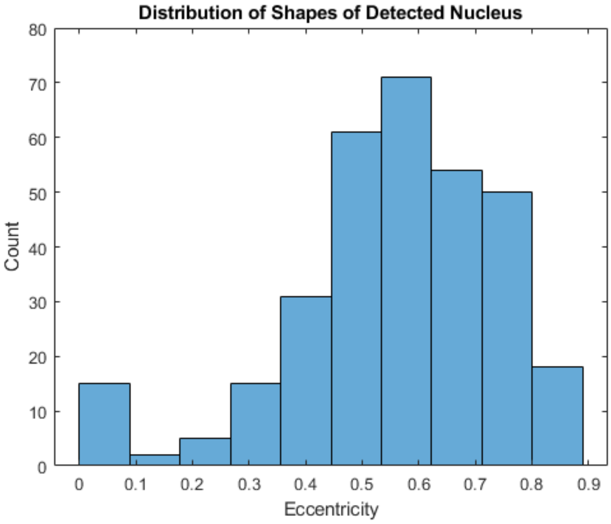
a) Image 1(StackNinja1.bmp)

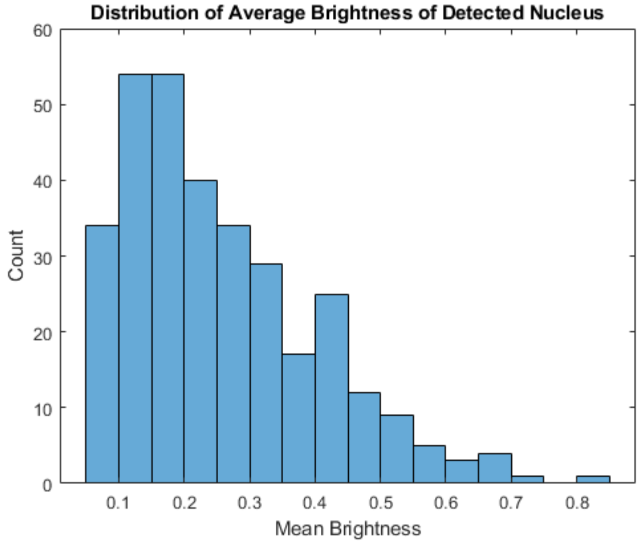
Binary Image Marking Regions Corresponding to Nuclei



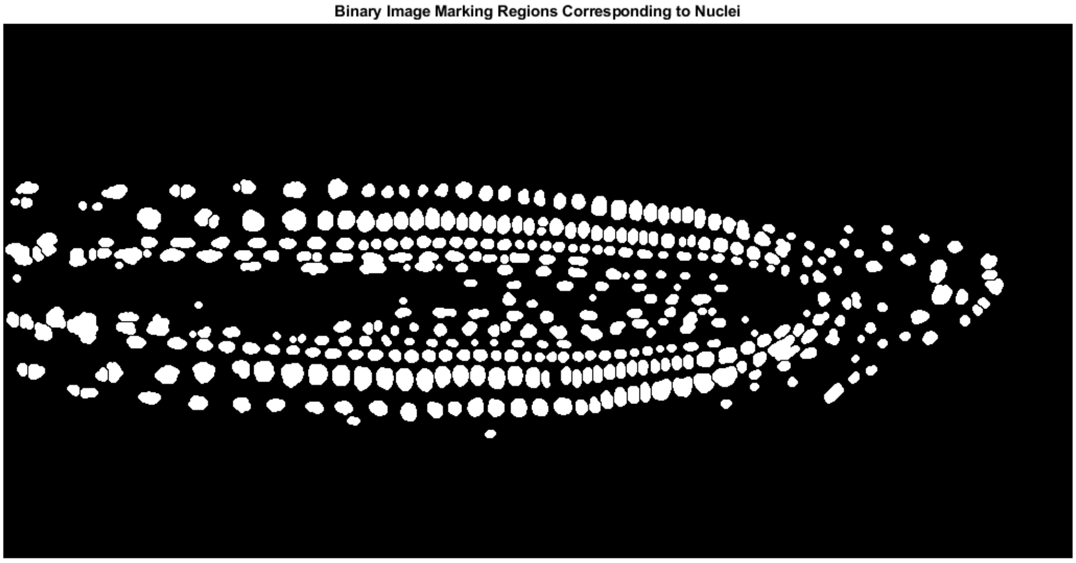
Total Nuclei Detected: 322



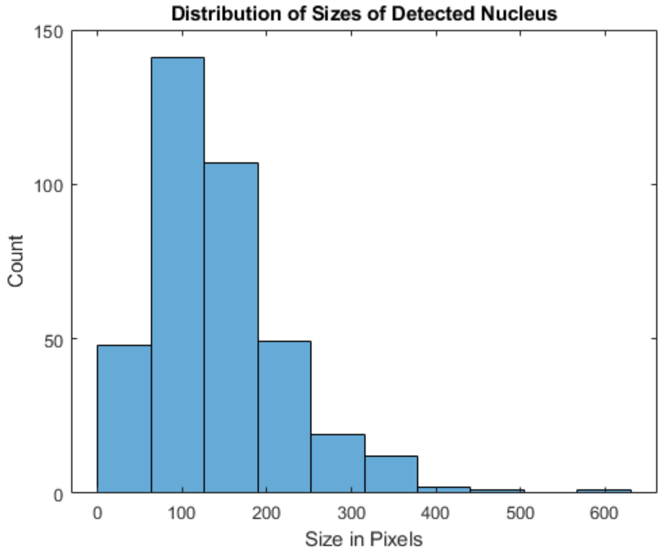


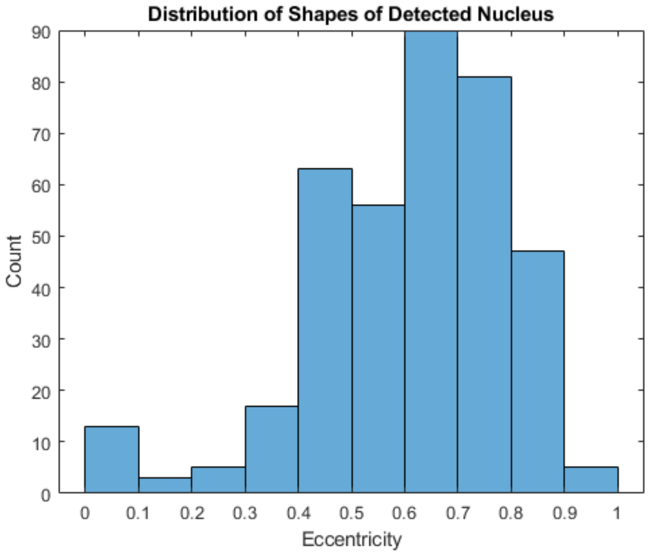


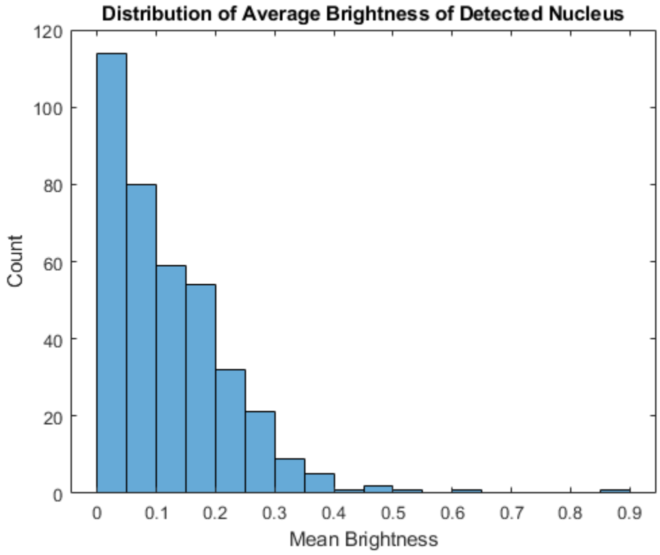
b) Image 2(StackNinja2.bmp)



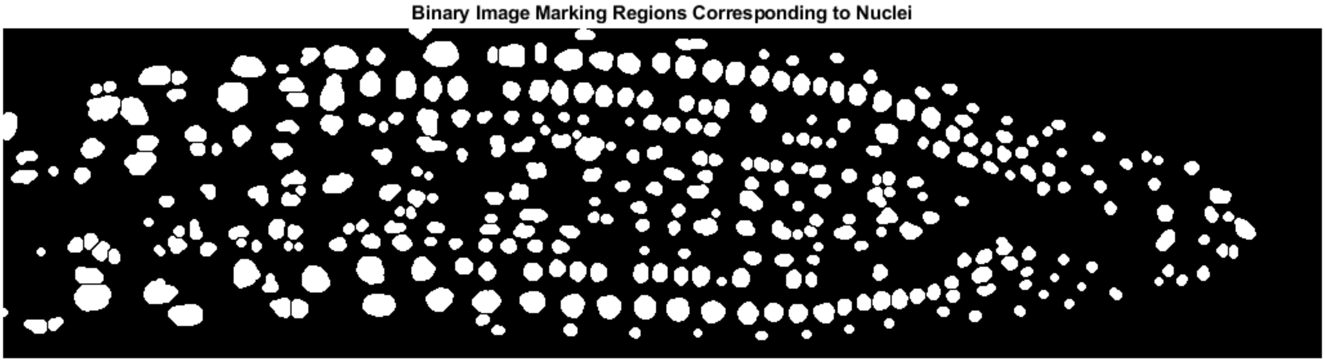
Total Nuclei Detected: 380



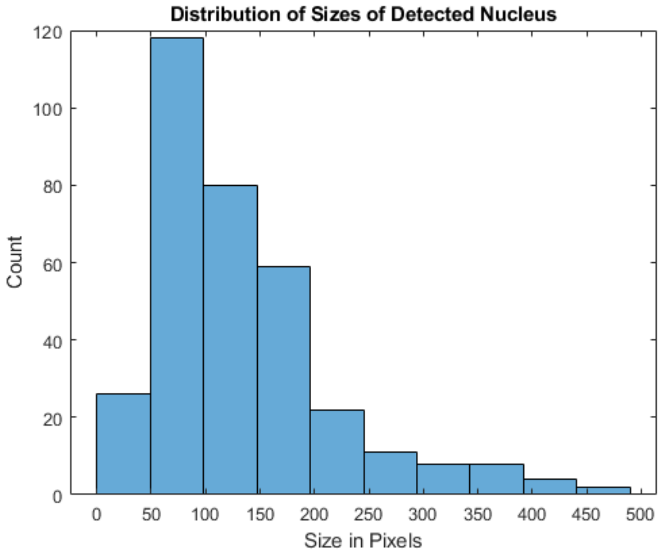


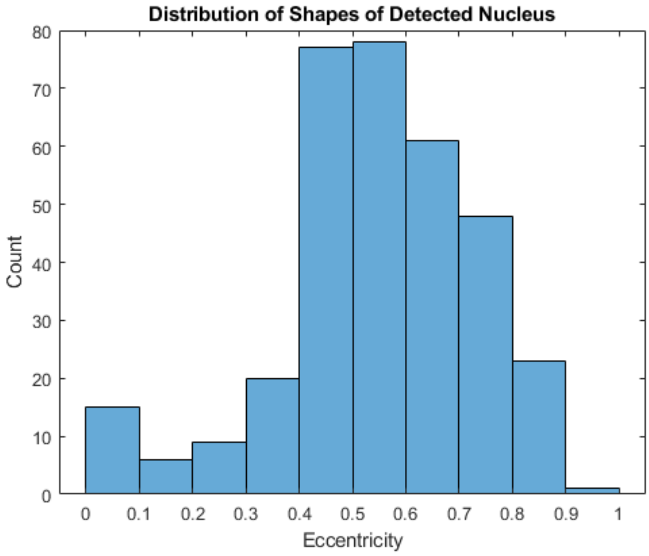


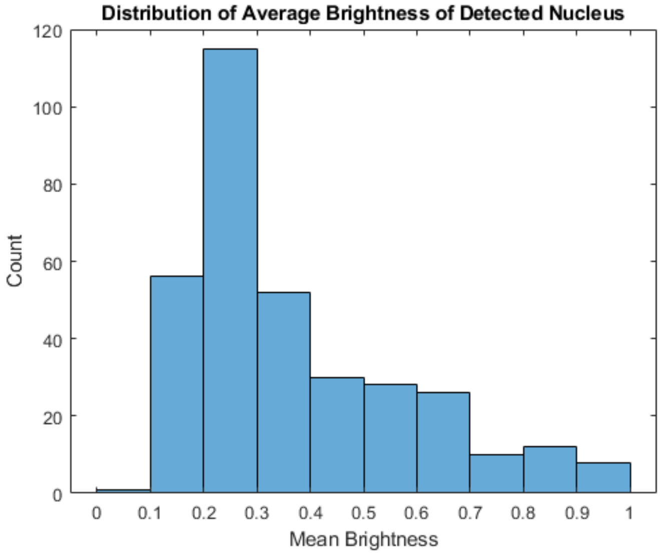
c) Image 3(StackNinja3.bmp)



Total Nuclei Detected: 338







Conclusion

Based on the results of analysis, the proposed pipeline methodology does have its pros and cons. This methodology performs exceptionally well when handling nucleus in reasonably bright areas. Besides, the watershed algorithm used in the methodology has improved the segmentation in most cases where two or more nuclei are being connected. However, there exist downsides for this methodology. For instance, it has difficulty in segmenting nuclei that exist in a relatively dark area. This has caused the formation of outliers in the analysis of nucleus size as two or more nuclei tend to connect with each other (inseparable), resulting in large number of pixels counts in a nucleus. Moreover, this methodology struggles to detect small-sized nucleus located in low brightness area. To improve, a suitable gamma correction to the brightness of input image may be applied, leading to a brighter image that has possibility to contain more noise formation that can be resolved by applying filtering techniques. In conclusion, despite having several downsides of this methodology. Still, it has done a great job in extracting and analysing cell nuclei detected from the supplied images.