COMP2032

Introduction to Image Processing

**Spring 2020/2021**

**Coursework Report**

**Extracting & Analysing Cell Nuclei**

**Name:** Sham Maatouk

**Student ID:** 20105652

Index

1. Abstract …………………………………………….…………………….........……...3

1. Introduction …………………………………………………….……..……...…....… 3

1. Methods …………………………………………………………...….…..….………. 4

3.1. Colour-Space Segmentation……………………..……………………..4

3.2. Producing a Binary Image………………………………………….......6

3.3. Noise Reduction………………………………………………………..8

3.4. Removing Redundant Regions…………………………………………9

3.5. Morphology……………………………………………………………10

3.6. Watershed Segmentation………………………………………………11

3.8. Count…………………………………………………………………..13

3.7. Image Analysis…………………………………………….…………..13

1. Results………………………………………………………………………………..14

4.1. Image 1: ‘StackNinja1.bmp’…………………………………………..14

4.2. Image 2: ‘StackNinja1.bmp’…………………………………………..19

4.3. Image 3: ‘StackNinja1.bmp’…………………………………………..24

1. Discussion……………………………… …………...…………………… 9

1. References ………………………………………………………….……. 10

1. Abstract

Confocal microscopy imaging has become increasingly important in life science research and medicine because of its ability to visualize the three-dimensional interiors of living cells and organisms. Automated image processing and analysis methods are now critical in this field to comprehend the complex organization of groups of interacting molecules within molecular machines and address fundamental biological issues prompted by molecular biology, optics, and technology developments.

This report discusses image processing and analysis techniques to detect, quantify, and analyse different attributes of plant cell nuclei obtained from a set of three confocal laser microscopic images.

1. Introduction

Confocal microscopy imaging is perhaps the most widely used optical technique in biological science today. This approach produces more detailed and desirable images than widefield microscopy. Its primary advantage over widefield microscopy is the absence of out-of-focus glare, which improves resolution and signal-to-noise ratio and the ability to collect serial optical parts of the specimen **[1].**It is also highly desirable for its efficiency and accuracy; hence, it is essential in developmental biology as it is a tool for interpreting growth dynamics**[2]**.

Several researchers have been developing automated methods for segmenting and counting cells in microscopic images, and it had been found that microscopic images are visualized and classified better through image processing techniques such as brightness correction and segmentation **[3].**

1. Methods

The solution for this problem was coded in MATLAB and utilized the Image Processing Toolbox. The following steps outline the workflow and details of the techniques used.

**3.1. Step 1: Colour-Space Segmentation**

**3.1.1. Separating Channels**

RGB colour space represents images as an m×n×3 matrix corresponding to red, green, and blue colour components of each pixel. This colour space is chosen to work with as the images provided had no other prominent colours than red, green, and light shades of yellow. Hence it would be more convenient to separate the colour channels and obtain the green one corresponding to nuclei. This method eliminates user-supplied parameters, as picking a colour-segmentation threshold value is not required, and it works accurately on all provided images.

The images are already in RGB format, so the three channels are seperated by:

im is the image matrix.

red= im(:,:,1);

green= im(:,:,2);

blue= im(:,:,3);

**3.1.2. Calculating Greenness**

After separating the channels, greenness identification is then formulated by:

greenness = green – (red + blue)/2.

This will generate a grayscale image showing the nuclei.

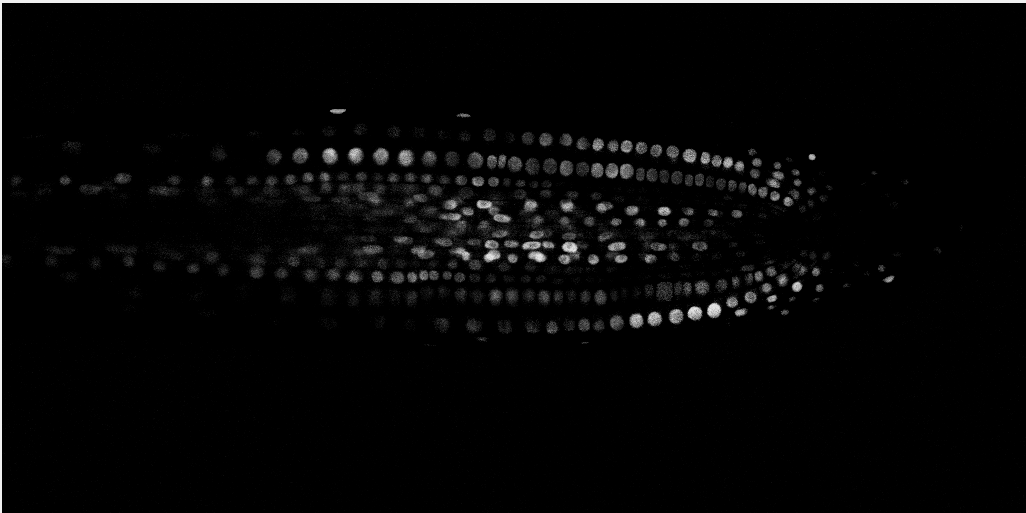


Figure 1

Original image:

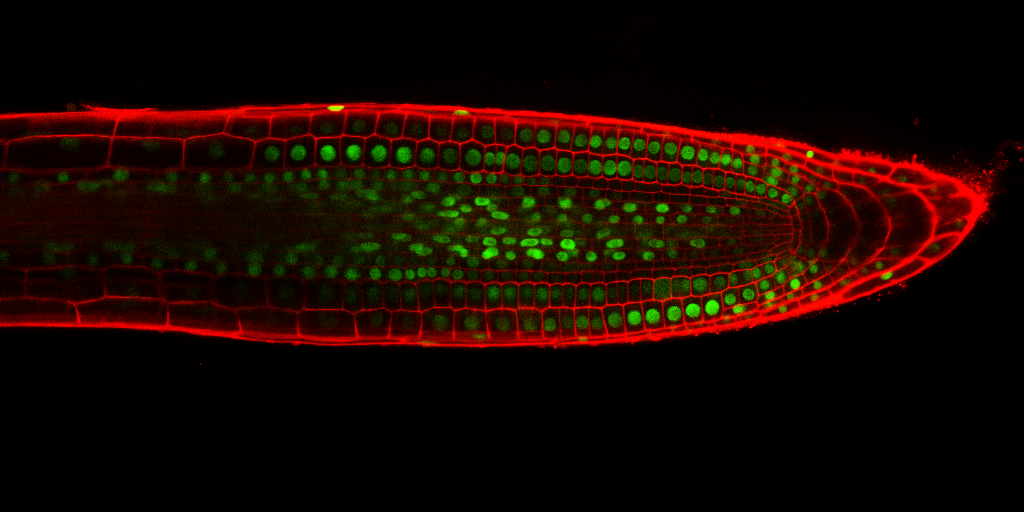


Figure 2

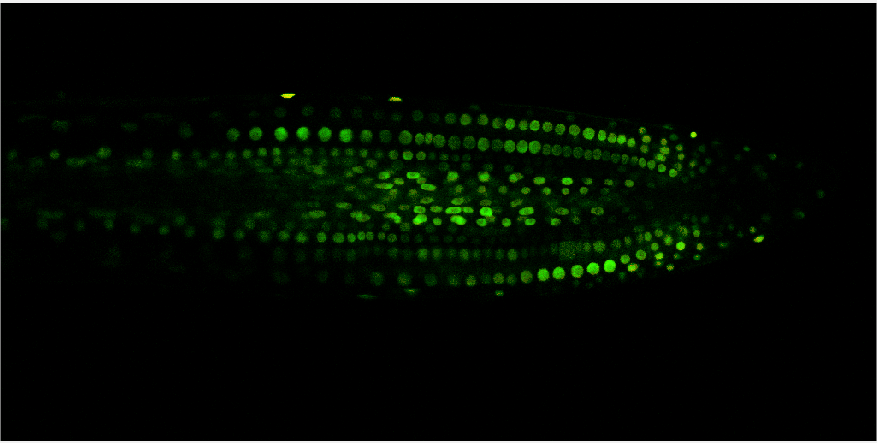
If we binarize the image (greenness) and invert it, then set every pixel element that correspond to 1 in greenness to 0 in the original image, an image that shows the nuclei clearly is produced, hence marking the first step as successful:

Figure 3

**3.2. Step 2: Producing a Binary Image**

**3.2.2. Adjusting Contrast**

As seen in Figure 1, some regions are more faded than others, giving us a low-contrast image; if this were to be binarized, some pixels corresponding to nuclei would be lost.

Hence adaptive-histogram-equalization technique is used. This technique enhances the contrast of the intensity by applying contrast-limited adaptive histogram equalization to the pixel values.

The method differs from histogram equalization as it works on smaller regions of the image, referred to as tiles, rather than on the entire image. The contrast of each tile is enhanced such that the output region's histogram closely matches the histogram defined by the 'Distribution' parameter; in this case, it is left as the default, which is 'uniform'. After removing artificially induced boundaries, adjacent tiles are combined using bilinear interpolation. Contrast can be limited, particularly in homogeneous areas, to avoid amplifying any noise present in the image.

This form of equalization is done by:

a = adapthisteq(s); (s is the image shown in Figure 1)

The output of a is shown by Figure 4, regions corresponding to nuclei that were faded in Figure 1 are now amplified:

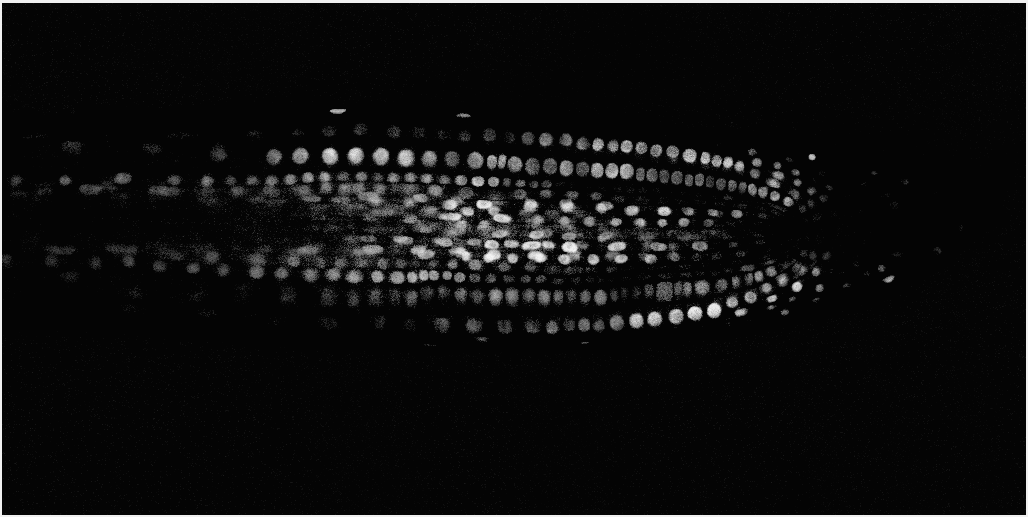


Figure 4

**3.2.2. Binarization**

The image in Figure 4 demonstrates a degree of nonuniform spatial illumination, and fixed thresholding would fail to produce a binary image that defines all the detected nuclei. Hence, the adaptive threshold algorithm is utilized. This method computes a different threshold value for every pixel that relies on a large-neighbourhood mean filter **[4]**.

The thresholding is done by:

bw = imbinarize(a,'adaptive');

Figure 5 shows the resulting binary image:

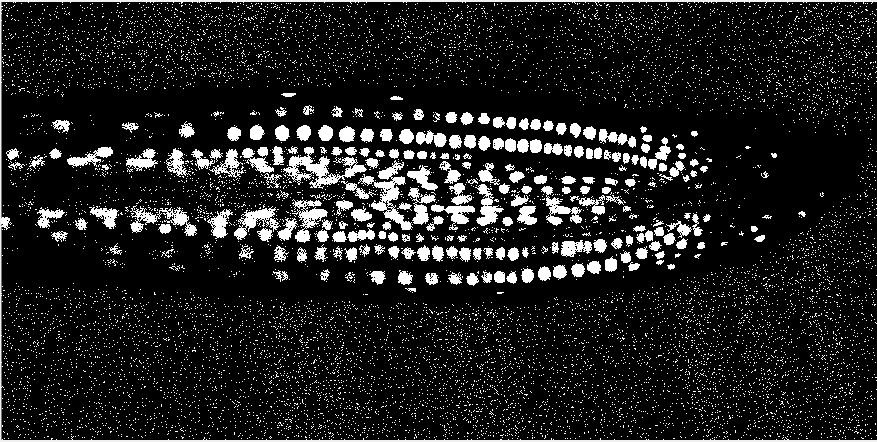


Figure 5

As shown in Figure 5, most of the regions corresponding to nuclei where detected, however, other background pixels are also set as foreground pixels producing noise, which will be handled in the next step.

**3.3. Step 3: Noise Reduction**

Median filters effectively reduce noise while also preserving edges by iteratively traversing the image, replacing each value with the median value of neighbouring pixels. The neighbouring pixel pattern is referred to as the "window", which slides pixel by pixel across the entire image. The median is determined by first sorting all the pixel values in the window numerically and replacing the currently selected pixel with the median pixel value.

Median filtering is done in the program by:

f=medfilt2(im); (im is the image shown in Figure 5)

This will produce a smoother image as shown in Figure 6:

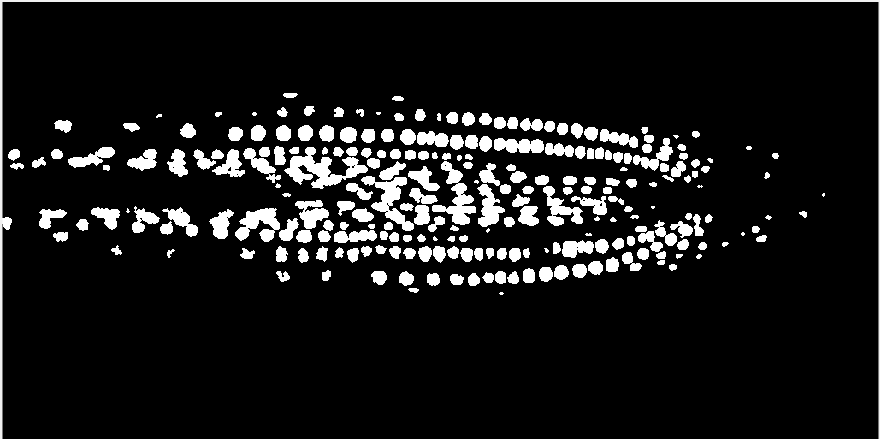


Figure 6

**3.4. Step 4: Removing Redundant Regions**

In some of the images there are still some noise after the filtering stage, hence all connected components that have fewer than 9 pixels are removed by:

b = bwareaopen(im,9);

Image before bwareaopen():

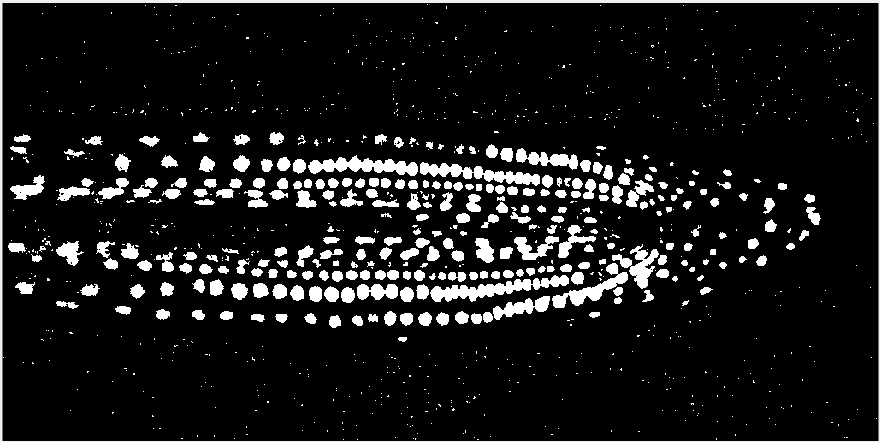


Figure 7

Image after bwareaopen():

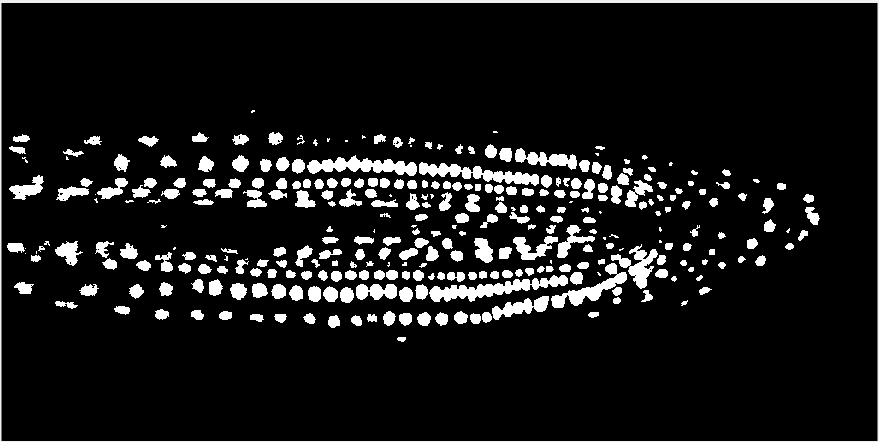


Figure 8

**3.5. Step 5: Morphology**

**3.5.1. Dilation**

As shown in Figure 8, some regions corresponding to nuclei that have low-intensity and are partially detected, are disconnected, and contain holes. A dilation operator is used to connect these areas; this expands the boundaries of objects by adding pixels. The output pixel's value is equal to the maximum of the values of all the pixels in its neighbourhood. A pixel is set to 1 if at least one of its neighbours has the value 1.

dilate = imdilate(b,true(2));

Before dilation After dilation

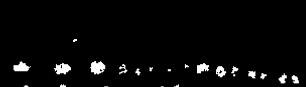


Figure 9 Figure 10

**3.5.2. Opening**

An opening is identified as an erosion followed by a dilation, both operations using the same structural element. An opening operator is applied to the images to remove small holes and define nuclei borders.

This is done by:

se = strel('disk',1); (the structuring element is a disk of 1 pixel neighborhood)

open = imopen(dilate,se);

After opening:

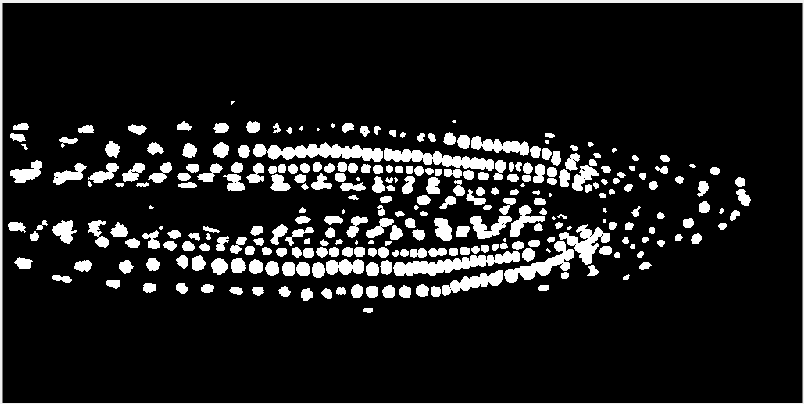


Figure 11

**3.6. Step 6: Watershed Segmentation**

As seen in Figure 11, some nuclei overlap with each other; this makes counting and image analysis inaccurate. To solve this issue, watershed segmentation is applied.

The algorithm considers the pixel values of user-defined markers as elevations, and floods basins from the markers until basins with different markers intersect along watershed lines. Usually, markers are chosen as the image's local minima, from which basins are flooded.

Overlapping nuclei are to be divided in Figure 12 below. To do so, an image that represents the distance to the background is computed. The maxima of this distance are chosen as markers, and flooding of basins from these markers divides the two circles along a watershed line.



Figure 12

The code below shows the implementation of the technique:

%Calculate the distance transform of the complement of the binary image.

D = bwdist(~im);

%Take the complement of the distance transformed image so that light pixels represent elevations

D = -D;

L = watershed(D);

% Set pixels that are outside the ROI to 0.

L(~im) = 0;

%convert to binary

L = double(L);

w = imbinarize(L);

The resulting segmented regions are shown in Figure 13:



Figure 13

**3.8. Count**

A total count of detected nuclei was obtained by boundary detection using the bwboundaries() function. bwboundaries() detects 8-connected contiguous white regions in a binary image.

The code is as follows:

[boundaries, label, num] = bwboundaries(im, 'noholes');

boundaries is a cell array where each element is the boundary of an object in the image. The boundary function computes the clockwise boundary of each object. Additionally, label is a labelled image computed by bwlabel. The number of labels in this image is returned in num, where the count is derived from.

**3.7. Image Analysis**

To analyse different attributes of the detected nuclei, regionprops() function is utilised to obtain pixel equivalent measurements of the different regions in the images. Given a labelled image, it automatically determines the properties of each labelled area. Below are descriptions of how the findings of each analysis were formulated. Refer to the results section for illustrations and findings.

**3.7.1. Size Analysis and Distribution**

regions = regionprops(label, 'Area', 'boundingbox');

'Area': is a scalar quantity and computes the number of pixels in each labelled area. Using this attribute, different aspects of the nuclei sizes can be concluded, such as the average nucleus size, distribution.

**3.7.2 Shape Analysis and Distribution**

stats = regionprops(label,'Area','Centroid','Eccentricity');

Considering that most nuclei exhibited ellipsoidal shape, the roundness of each nucleus is determined by its eccentricity, which is the ratio of the distance between the foci of the ellipse and its major axis length. Eccentricity values range from 0-to-1. An eccentricity value that is approximately close to 0 belongs to a circular object. Hence the roundness of the objects is determined by how close their eccentricity value is to 0.

Although the circularity property can be obtained from regionprops(), for this case, the eccentricity is deemed more accurate as circularity is not recommended for small objects, since the results might exceed the circularity value for a perfect circle.

**3.7.3 Brightness Analysis and Distribution**

stats = regionprops(label,im,'MeanIntensity');

The brightness of each nucleus is determined by considering the mean intensity of its region. Regions corresponding to high intensity values are ones with higher brightness, others with low intensity values have lower brightness.

1. Results.

**4.1. Image 1: ‘StackNinja1.bmp’**

**4.1.1. Nuclei Detection**

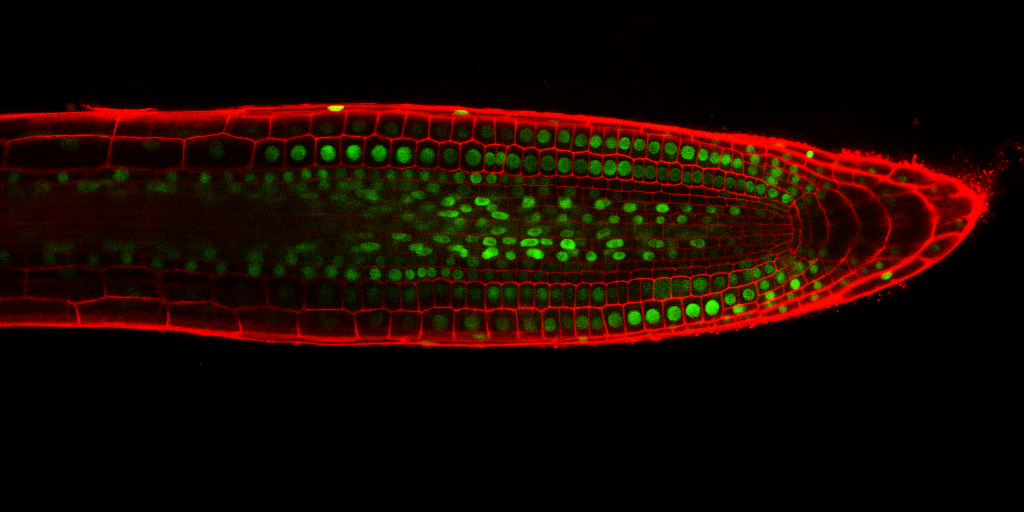


Figure 14: original image

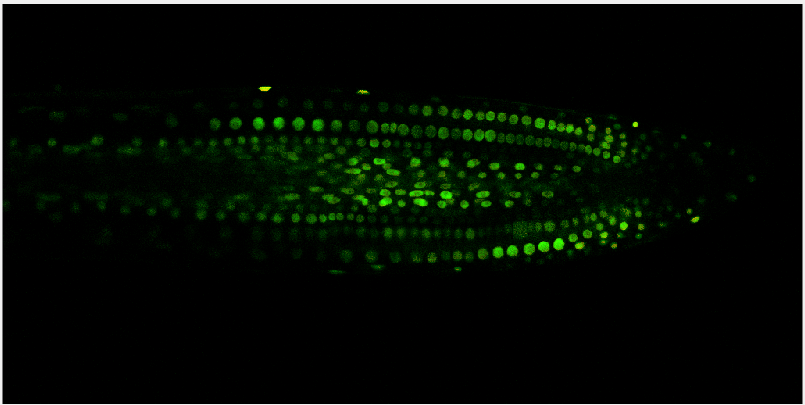


Figure 15: Image of isolated nuclei

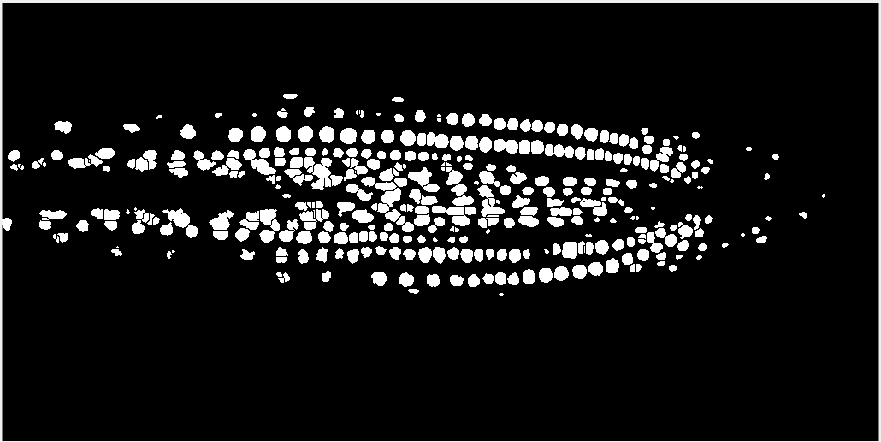
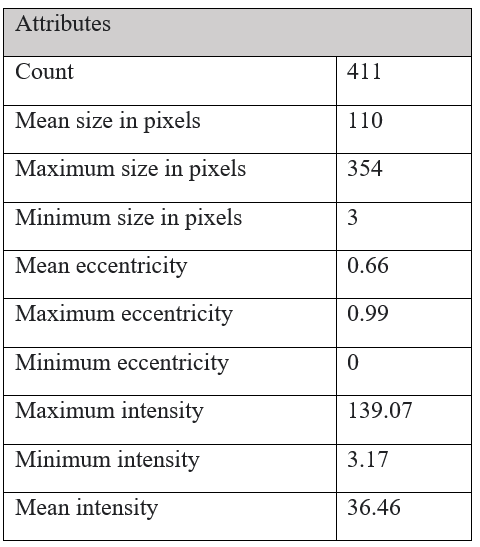
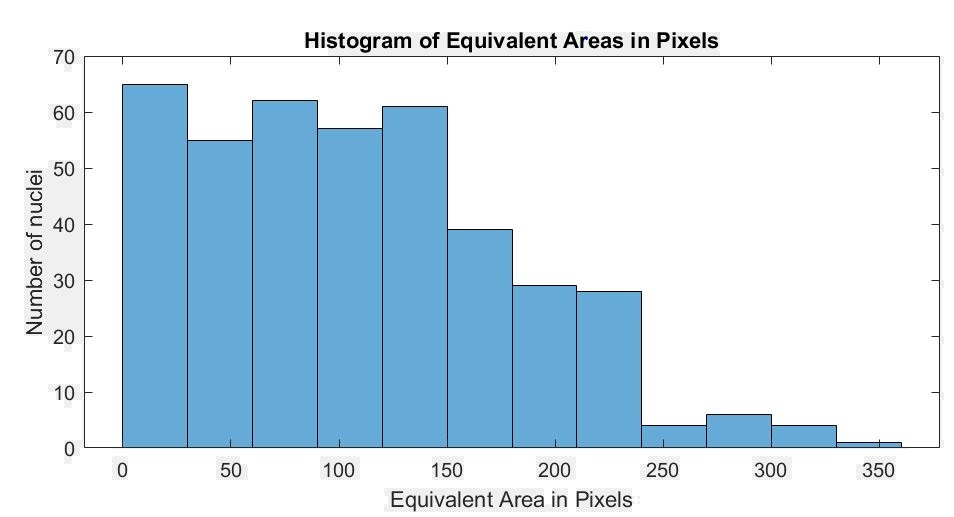


Figure 16: Binary image with detected nuclei positions



**4.1.2. Size Analysis**



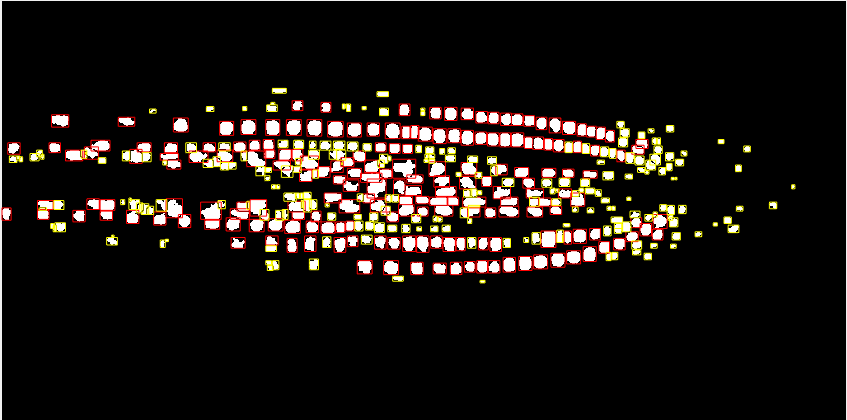
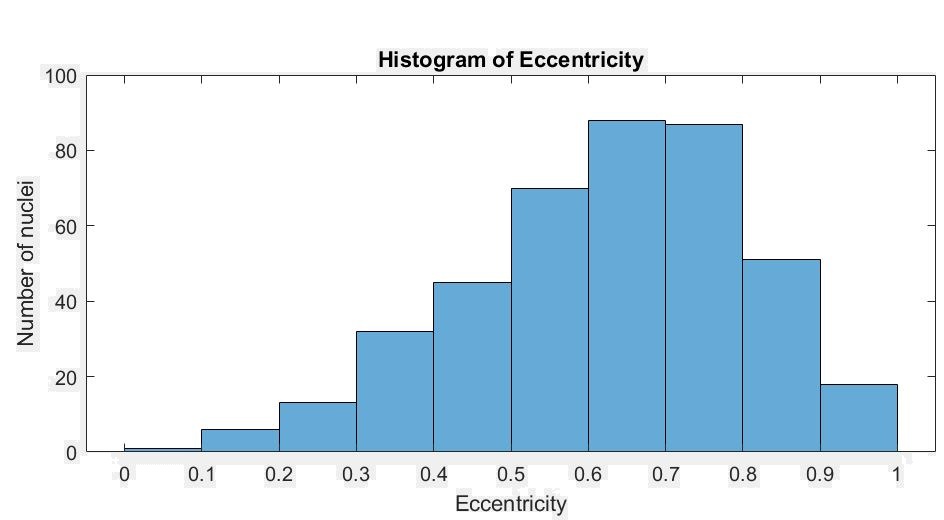


Figure 18: nuclei bigger than average are bordered in red, others in yellow.

**4.1.3. Shape Analysis**



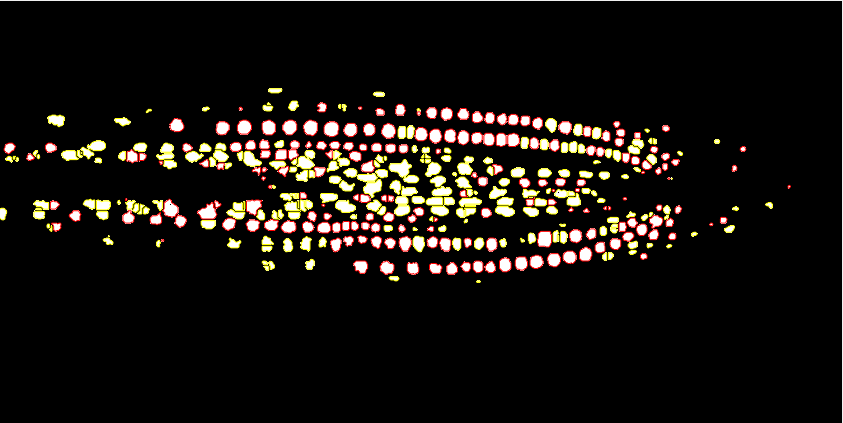
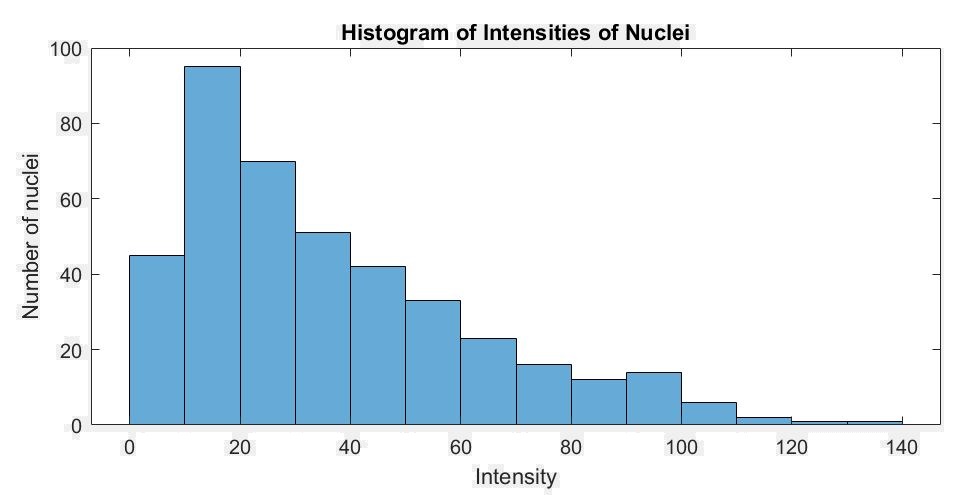


Figure 20: nuclei rounder than average are bordered in red, others in yellow

**4.1.4. Brightness Analysis**



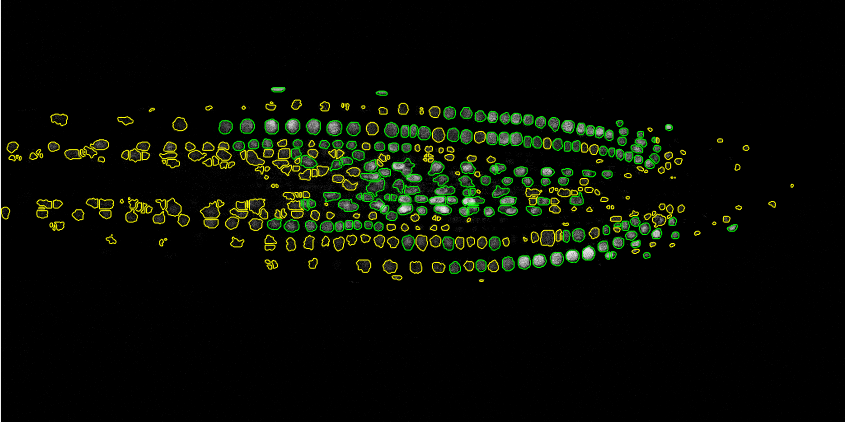


Figure 22: nuclei brighter than average are bordered in green, others in yellow.

**4.2. Image 2: ‘StackNinja2.bmp’**

**4.2.1. Nuclei Detection**

A picture containing invertebrate, worm, indoor, lit

Description automatically generated

Figure 23: Original image

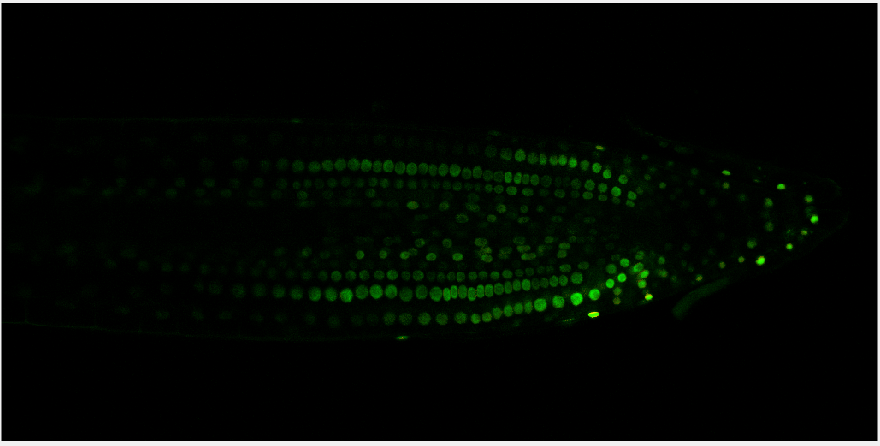
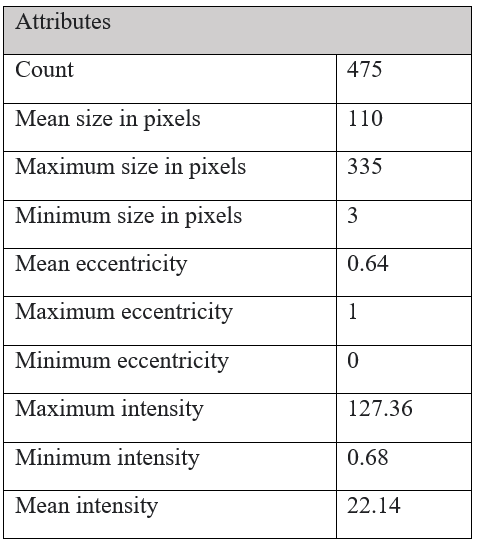


Figure 24: Image of isolated nuclei



Figure 25: Binary image with detected nuclei positions



**4.2.2. Size Analysis**



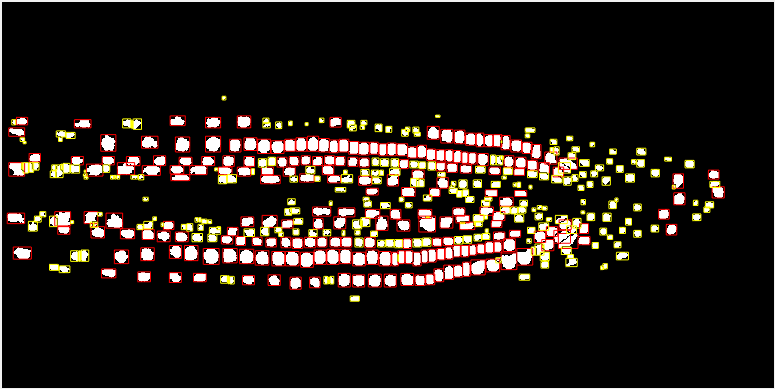
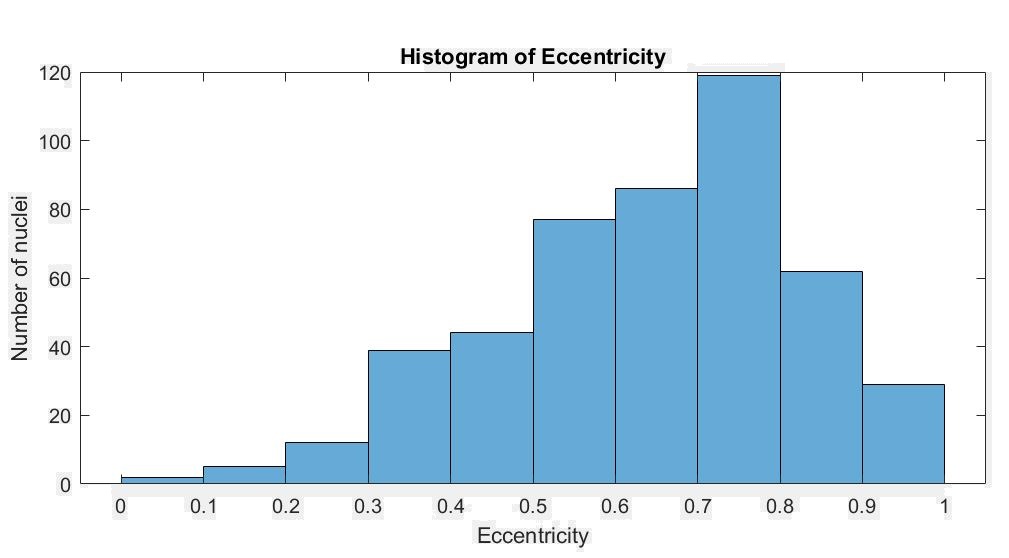


Figure 27: nuclei bigger than average are bordered in red, others in yellow.

**4.2.3. Shape Analysis**



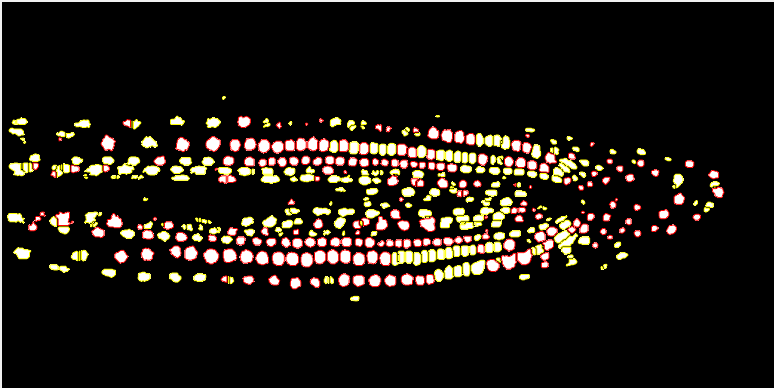
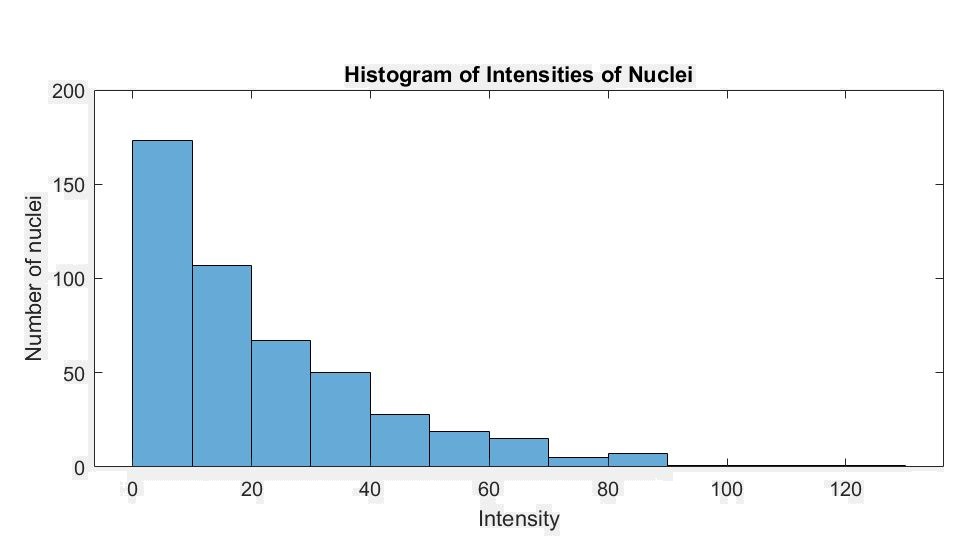


Figure 29: nuclei rounder than average are bordered in red, others in yellow

**4.2.4. Brightness Analysis**



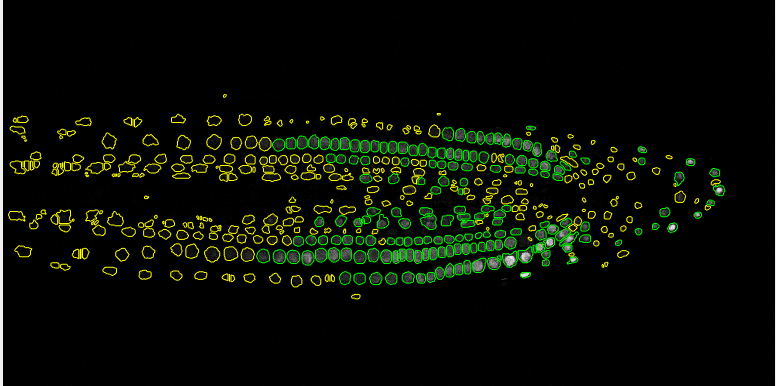


Figure 31: nuclei brighter than average, others in yellow

**4.3. Image 3: ‘StackNinja3.bmp’**

**4.3.1. Nuclei Detection**

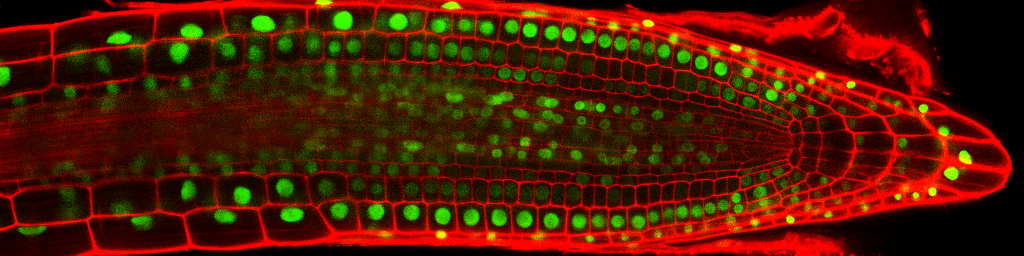


Figure 32: Original image

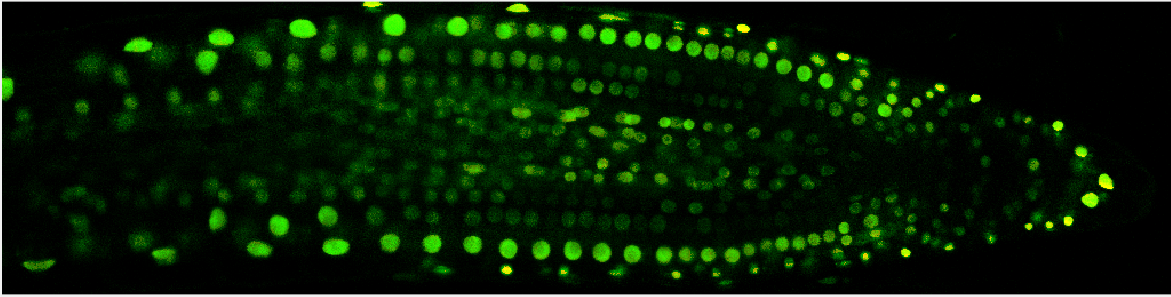


Figure 33: Image of isolated nuclei

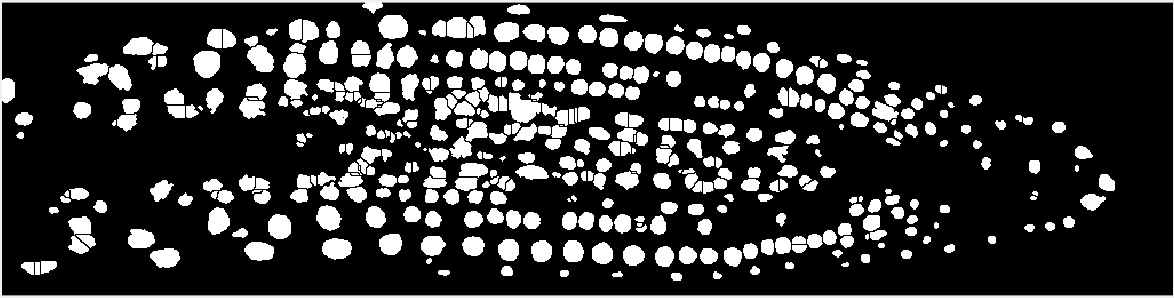
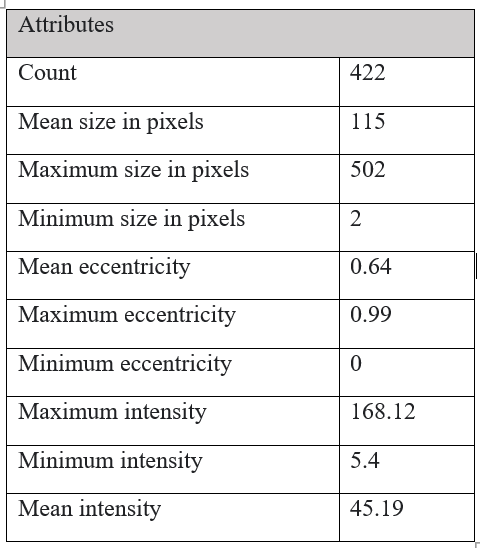
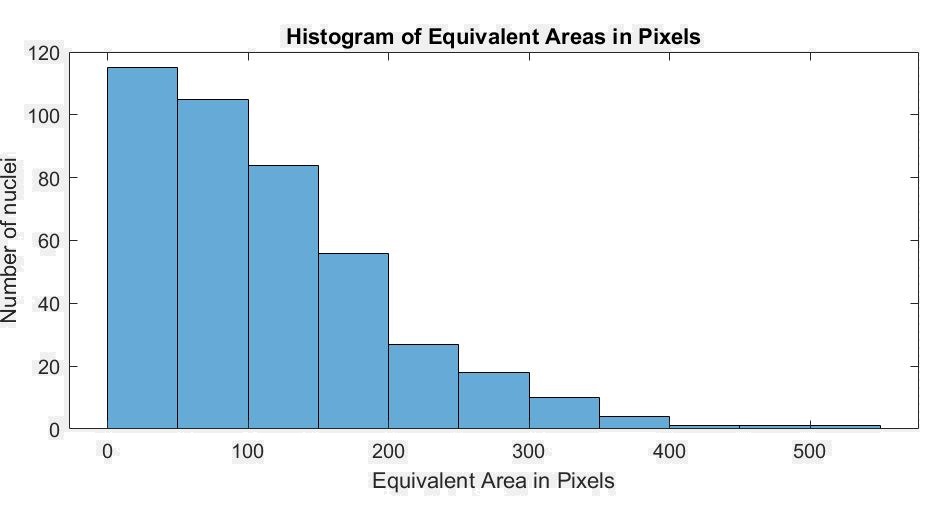


Figure 34: Binary image with detected nuclei positions



**4.3.2. Size Analysis**



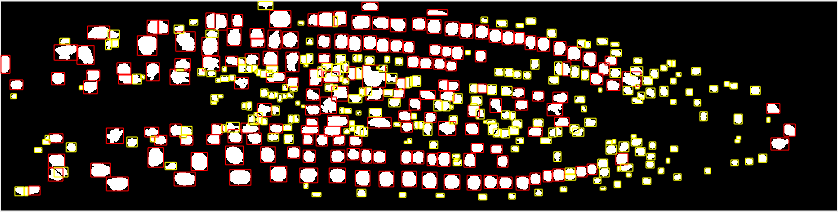
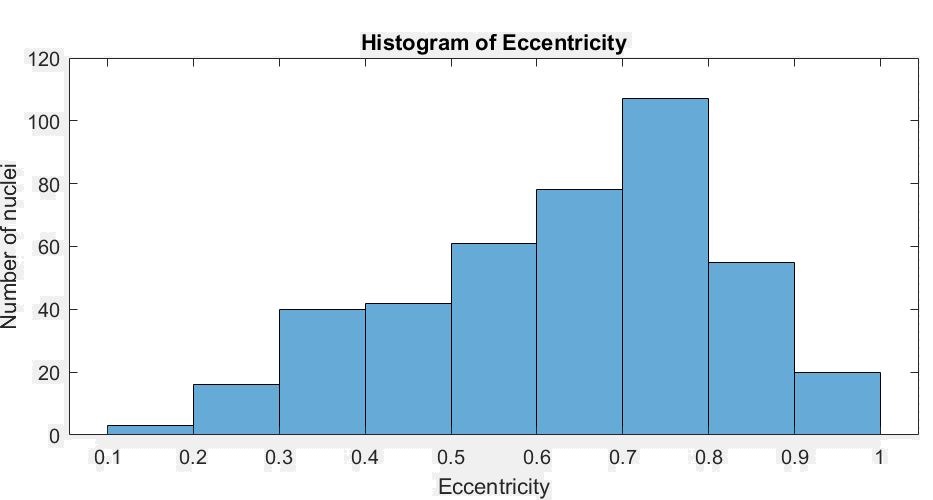


Figure 36: nuclei bigger than average are bordered in red, others in yellow

**4.3.3 Shape Analysis**



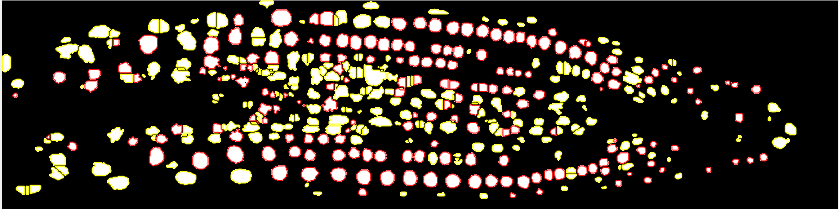
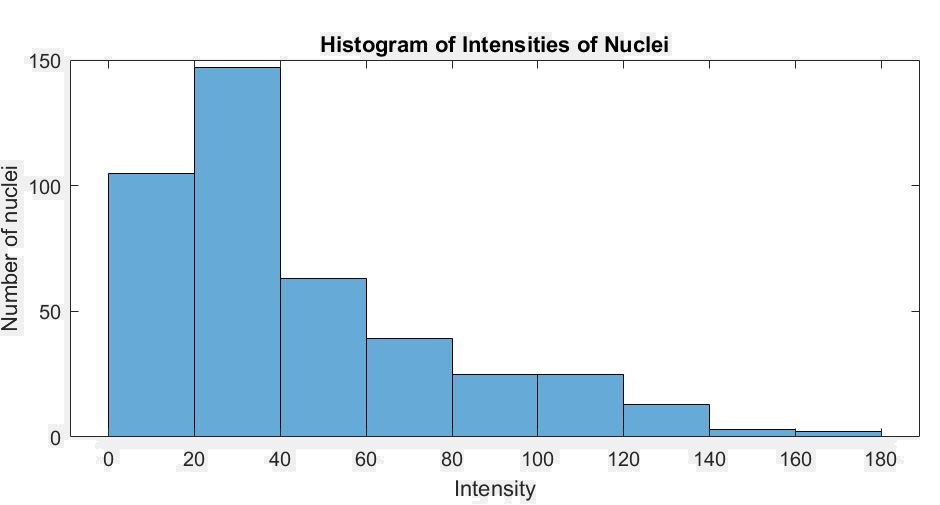


Figure 38: nuclei rounder than average are bordered in red, others in yellow

**4.3.4. Brightness Analysis**



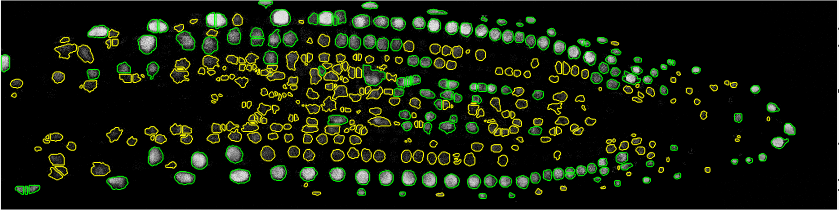


Figure 39: nuclei brighter than average are bordered in green, others in yellow.

1. Discussion

Overall, most nuclei are successfully detected in all three images and analysed, however due to specific drawbacks in some employed techniques the results are not highly accurate.

**5.1. Strengths**

Colour-space segmentation performs excellently on all three-image, leaving no red borders, and defining all green nuclei. Moreover, the adaptive thresholding and equalization techniques conserve data by producing images with white areas corresponding to almost all nuclei, regardless of spatial illumination and low contrast. Lastly, applying the median filter instead of mean or Gaussian filters eliminates salt-and-pepper noise while also maintaining edges.

**5.2. Weaknesses and Improvements**

Some improvements could be taken in morphological processing and segmentation to acquire more accurate results and conserve data.

Although morphological dilation improves the quality by closing holes and defining nuclei, it disturbs the true shape and size of detected nuclei, leading to slightly inaccurate measurement values. Furthermore, the extra opening stage causes the loss of some smaller nuclei, which leads to an inaccurate count. Finally, watershed segmentation has an over-segmenting effect on some nuclei regions, further affecting count, and shape and size analysis.

1. References
   1. Haimovich, G. (2012, 29th of April). Basics in Confocal microscopy and image analysis. [Weblog]. Retrieved 22 April 2021, from <https://greenfluorescentblog.wordpress.com/2012/04/29/basics-in-confocal-microscopy-and-image-analysis/>
   2. Mkrtchyan, K., Singh, S., Liu, L, Venugopala Reddy, G., & Chowdhury,A.,K. (2011, September). Efficient cell segmentation and tracking of developing plant meristem*.* University of California, Riverside.
   3. Lim, K., Park, S. H., Kim, J., SeonWoo, H., & Choung, P. (2013). Cell Image Processing Methods for Automatic Cell Pattern Recognition and Morphological Analysis of Mesenchymal Stem Cells - An Algorithm for Cell Classification and Adaptive Brightness Correction -. Journal of Biosystems Engineering. Retrieved 22 April 2021, from <https://doi.org/10.5307/JBE.2013.38.1.055>
   4. Bradley, D., & Roth, G. (2007). Adaptive Thresholding using the Integral Image. Journal of Graphics Tools, 12(2), 13–21. <https://doi.org/10.1080/2151237x.2007.10129236>