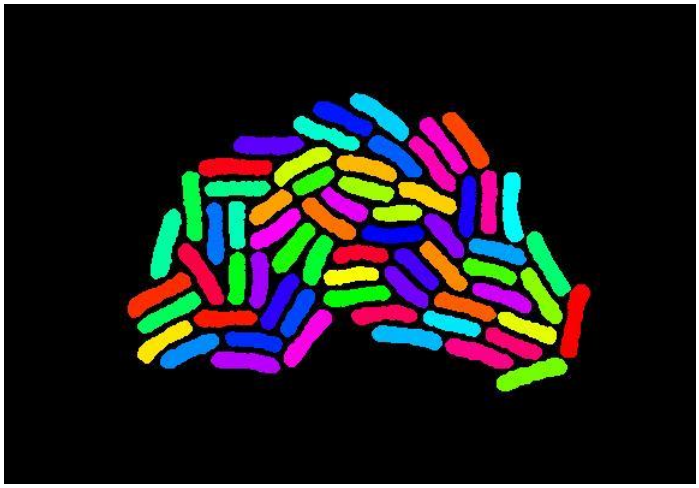
Image segmentation of bacteria in time-lapse movies

August 31, 2015

Manoj A Kurien [1, 2], Bruno M.C Martins [2], Christian Schwall [2] and James C.W Locke [2]

[1] Undergraduate Research Opportunities Programme, Cambridge University Engineering Department

[2] Sainsbury Laboratory, University of Cambridge



Abstract

The adoption of new computational tools is revolutionising cell biology and making it a true quantitative science. Imaging techniques allow us to probe directly into living cells, and to extract information about each individual cell. We can then use this information to build mathematical models and make testable predictions. Some of the main challenges are the identification of individual cells and tracking individual cells across time and space. The Locke Group at the Sainsbury Laboratory has been developing custom-made tools to address these challenges.

This UROP project investigates four different methods to improve the current software (Schnitzcells) used by the Locke Group for the segmentation of S. elongatus. The methods developed here - Watershed method, Marker watershed method, Gamma correction and threshold method and Hybrid method are tested on four different time-lapse movies and their performance scrutinized against Schnitzcells. The Hybrid method tested produces improved image segmentation. We also investigate tools to study the fluorescence in the field of view which reveals non-uniform fluorescence should be considered in further analysis and studies.

Contents

1 Introduction

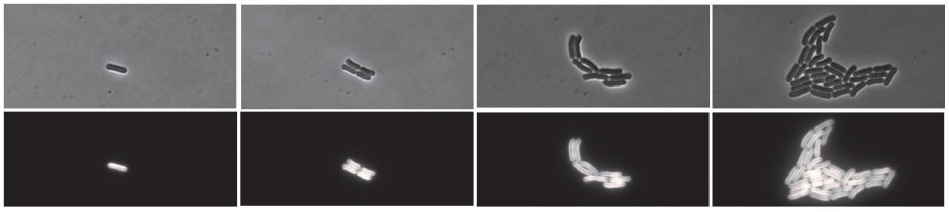
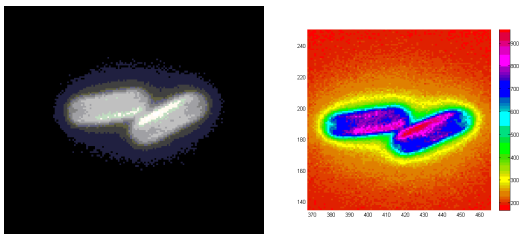


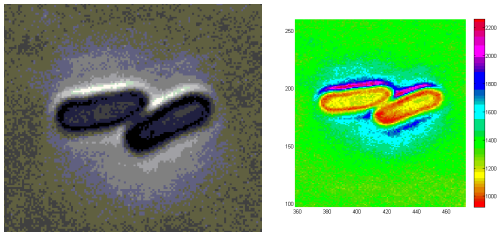
Figure 1: Phase-contrast (above) and red fluoresence channel (below) time-lapse images of cyanobacteria in 1.5 day time steps

Studying gene expression at the single cell level with time-lapse microscopy enables the analysis of gene circuit dynamics. Gene expression is studied by analysing the expression of fluorescent protein markers which emit light when excited by a particular wavelength of light. The protocol involves seeding and growing bacteria on small agarose pads, imaging the microcolonies and using MATLAB code to segment and track cells frame by frame [1]. Good segmentation and tracking is essential for accurate measurements and characterization of bacterial gene dynamics. In this project, we focus on the image segmentation of rod- shaped cyanobacteria (*S. elongatus*). Time-lapse images of both phase-contrast imaging and red fluorescence (RFP) channels are taken at a period of 45 minutes for cyanobacteria.

Phase-contrast imaging works with the principle that different structures have different refractive indices. The bending of light as it passes through the structure results in phase differences between the light waves which can be processed to form images. *S. elongatus* also manufactures a protein which when excited by a particular wavelength of light, the excited electrons relax to their ground state by emitting photons in the red fluorescence channel. These emitted photons can be detected to produce the red fluorescence channel images (RFP). Due to the non-uniform luminance of the cells, image segmentation is non-trivial (see figure (2)). The Locke lab uses MATLAB code called Schnitzcells [1] for the image analysis of bacteria. It is optimized for image segmentation and tracking of *B. subtilis*.



(a) RFP channel image (b) RFP channel image intensity surface



(c) Phase-contrast image (d) Phase-contrast image intensity surface

Figure 2: The RFP channel image and phase-contrast image both illustrate non-uniform intensities within and surrounding cells (as seen in the intensity contours plot). This uneven luminance makes accurate segmentation a challenge. (Movie 1 frame 5)

2 Problem definitions

2.1 Measurement of fluorescence in field of view

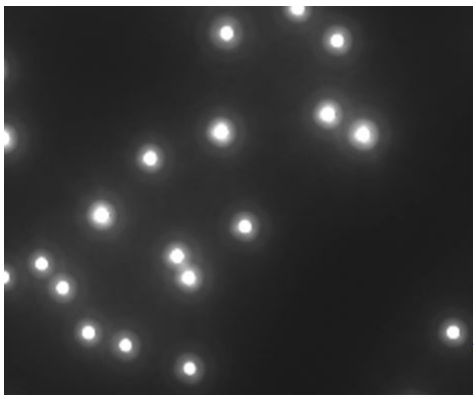


Figure 3: Fluorescent beads on an agar plate used to measure the fluorescence in field of view.

The light intensity distribution on the image plane in the setup is a factor that has to be considered for analysing gene expression. To study this, fluorescent beads (size) can be placed on the agar plate and imaged. However, due to the random distribution of beads, they can be too close to each other or touching. In such situations, fluorescence would interfere with each other's, producing much higher intensities. Such beads would have to be ignored when processing a heat-map to study the distribution of fluorescence on the focal plane.

The heat-map would reveal the fluorescence in the field of view and these results can be used as a correction factor on the fluorescence measured from bacteria, test different cameras and improve the overall experimental setup. Another method that could be used to measure the fluorescence is to coat the entire agar plate in fluorescent marker before imaging. However, this tends to produce bright, over saturated images. Hence, image processing is required to segment beads from images, exclude unsuitable beads and then average the intensities of many images to produce a heat-map.

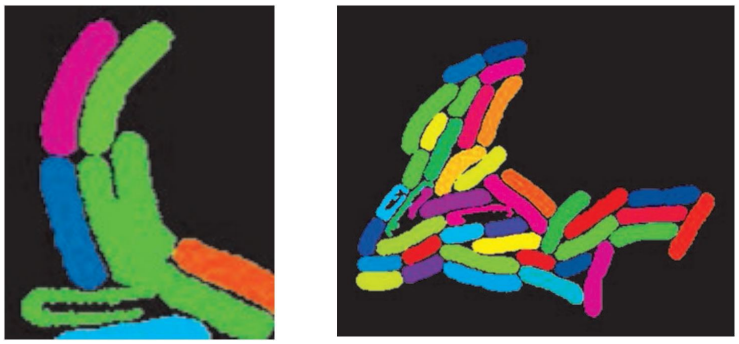
2.2 Segmentation

Current code for segmentation (Schnitzcells) of cyanobacteria results in frequent errors such as missing cells, merged cells and incomplete cells. The code uses images of natural fluorescence in the RFP channel. The aim is to improve these segmentation issues.

The movies that were tested for improving this were movies:

1. Movie 1[1] - The errors with Schnitzcells here are missing cells and merged cells.
2. Movie 2[2] - Schnitzcells performs very well with this set, with occasional incomplete cells.
3. Movie 3[3] - The problems here are incomplete cells and merged cells.
4. Movie 4[4] - This movie has mutant bacteria, where RFP fluorescence is clustered at the ends of cells. Schnitzcells segmentation causes a few incomplete, missing and merged cells.

[1] *2015-03-04-Bruno-Cyano2 (PSB7-11_1)*
[2] *2015-03-11-Bruno-Cyano2 (PSB7-16_1)*
[3] *2015-03-18-Bruno-Cyano2 (PSB-01_1)*
[4] *2015-03-04-Bruno-Cyano2 (PSB7-13_1)*



(a) Merged cells and incomplete cells.

(b) Incomplete cells.

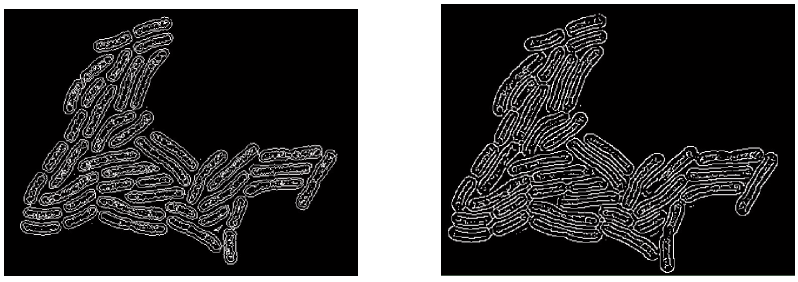
Figure 4: Segmentation errors with Schnitzcells (Frames 126 and 161 from Movie 3)

The incomplete cell shapes arise due to incomplete edge detection in Schnitzcells. The merged cells arise due to inaccurate filling of closed shapes from the edge detection. Missing cells also arise from incomplete edge detection. The fundamental difficulty in edge detection are due to the 'bleed through' of RFP fluorescence, non-uniform fluorescence within a cell and noise. Moreover, there are slight differences between various movies in their focus and experimental setup which lead to varying image segmentation performances across movies.

3 Image processing tools

3.1 Edge detection

Edge detection operates by looking for sharp changes in pixel intensity over an image. Edge detection can be performed with MATLAB's image processing toolbox function `edge`. It supports various methods of edge detection, different options such as threshold and standard deviation of the edge filters. The *Canny* method and the *Laplacian of a Gaussian (LoG)* method produced the most promising results for edge detection (fully closed edges outlining a cell). The LoG method looks for zero-crossings after spatial filtering the image with a LoG filter. This is the method used by the current version of Schnitzcells and produced the most suitable edges when the threshold and standard deviation were tuned. Hence, this is the method that has been opted for edge detection.



(a) Edge detection with LoG method.

(b) Edge detection with Canny method.

Figure 5: More closed edges are detected with LoG method (Frame 161 Movie 3, Threshold 0 and sigma = 0.9)

3.2 Morphological operations

These operations process images based on their shape and connectivity. They apply a structuring element to an input image to produce an output image processed by comparing the pixels to its neighbours. The two fundamental morphological operations are dilations (adding pixels to object boundaries) and erosions (removing pixels from object boundaries). They can be performed on grayscale images with the set of

functions such as `imopen`[5] and `imclose`[6] and on binary images with `bwmorph`. They are used in the segmentation process to affect the overall shape, size and connectivity.

3.3 Thresholding

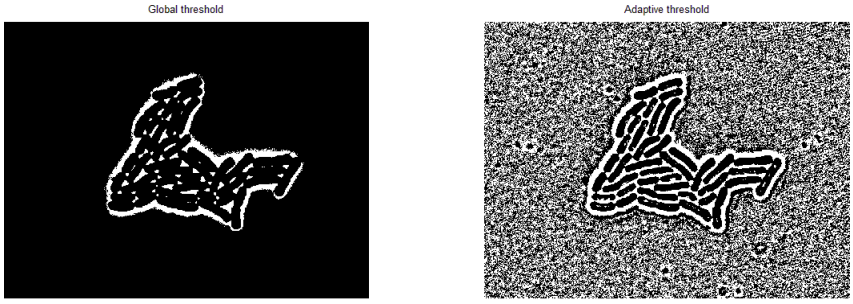


Figure 6: Global thresholding and adaptive thresholding. Adaptive thresholding produces better cells, but requires removal of the background noise. (Phase image 161 Movie-3)

3.3.1 Global threshold

Transforming grayscale images acquired from phase and RFP channels to binary images requires the selection of a threshold. Initially, the function `im2bw` with the function `graythresh`[7] was used for this process which selects a threshold for the entire image i.e. a global threshold. The function `graythresh` implements the Otsu algorithm for thresholding [5].

3.3.2 Adaptive threshold

Adaptive thresholding varies the threshold (within a specified window size) over the image. This improves the thresholding process by taking into account non-uniform illumination. The algorithm for adaptive thresholding (`adaptivethreshold`) was downloaded from MATLAB central file exchange [2]. In figure (6), we can see the much improved performance of adaptive thresholding of the phase image in segmentation. However an additional step is required to remove the background noise.

3.4 Gamma correction

Gamma correction enables non-linear contrast adjustment of grayscale images. "Gamma can be any value between 0 and infinity. If gamma is 1 (the default), the mapping is linear. If gamma is less than 1, the mapping is weighted toward higher (brighter) output values. If gamma is greater than 1, the mapping is weighted toward lower (darker) output values" [3]. This relationship can be seen in figure (7). Non-linear contrast adjustment prior to thresholding enables better image segmentation. This contrast adjustment is performed with the MATLAB function `imadjust`.

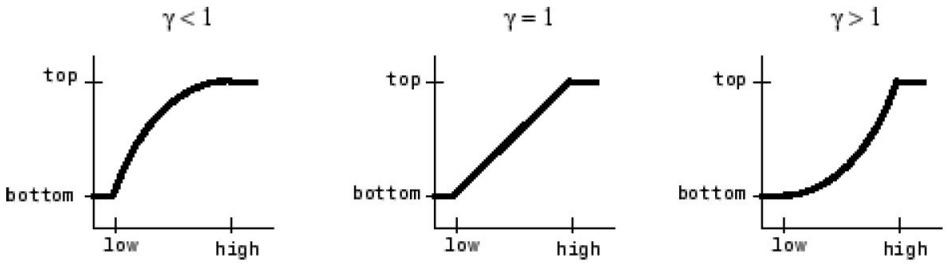


Figure 7: "The three transformation curves show how values are mapped when gamma is less than, equal to, and greater than 1. (In each graph, the x-axis represents the intensity values in the input image, and the y-axis represents the intensity values in the output image)" [3]

[5] Morphological opening
[6] Morphological closing
[7] Takes in a grayscale image and automatically selects a global threshold to output a binary image.

3.5 Histogram adjustment

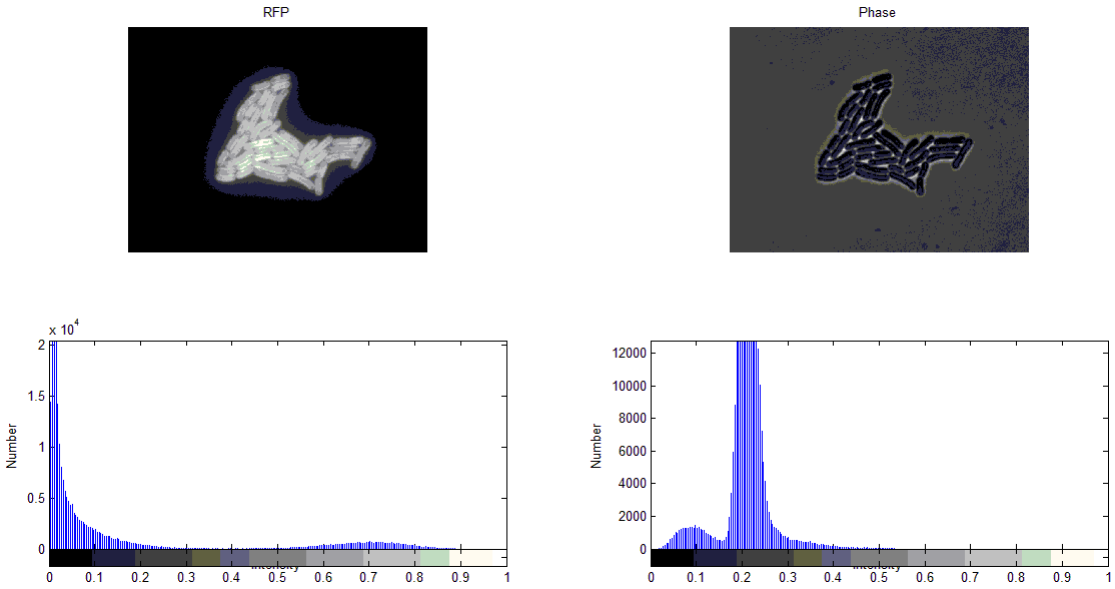


Figure 8: Typical image histogram for RFP and phase images. (The intensities have been normalised)

An image histogram displays the distribution of pixel intensities in a grayscale image. The image histogram can be studied to select appropriate image enhancement operations. "Histogram equalization involves transforming the intensity values so that the histogram of the output image approximately matches a specified histogram" [3].

Histograms could be adapted to fit a uniform, exponential or Rayleigh distribution (with a tuning parameter (α) for the shape of the exponential and Rayleigh distributions) with MATLAB's Adaptive Histogram Equalization function. In addition, the box size within which histogram equalization is performed can also be tuned. This histogram adjustment is performed with the MATLAB function `adapthisteq`.

3.6 Filtering

3.6.1 Standard deviation filter

The standard deviation filter is a form edge detection as it highlights regions where the standard deviation is large. It returns the standard deviation value of the 3-by-3 neighbourhood around each pixel. This filtering is performed with the MATLAB function `stdfilt`.

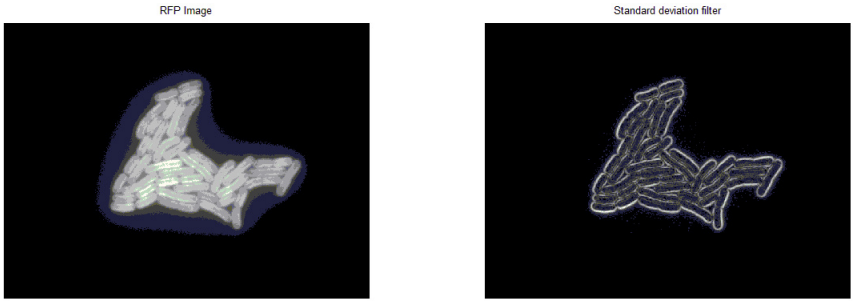


Figure 9: RFP image and the standard deviation filter.

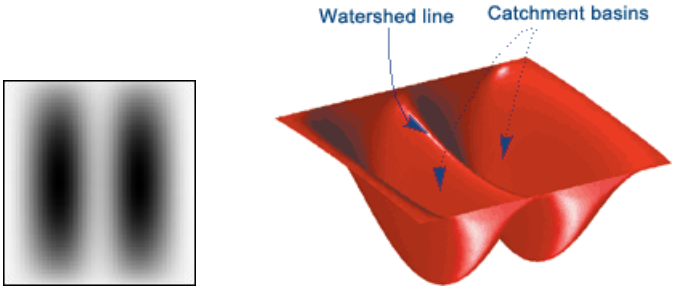
3.6.2 Adaptive filter

Standard Gaussian smoothing for noise reduction leads to blurring of edges as well. This is unsuitable for accurate edge and shape detection. An adaptive filter identifies 'significant edges' and performs smoothing without blurring sharp edges. "The `wiener2` function applies a Wiener filter (a type of

linear filter) to an image adaptively, tailoring itself to the local image variance. Where the variance is large, `wiener2` performs little smoothing. Where the variance is small, `wiener2` performs more smoothing" [3].

3.7 Watershed

This tool enables image segmentation by utilizing a topographic interpretation of the image intensity. 'Water' fills up from a regional minimum and forms boundaries (watershed lines) where they meet other regional minima. This process is performed with the MATLAB function `watershed`.



(a) Grayscale image

(b) Topographic image with catchment basins and watershed lines

Figure 10: The watershed transform. [6]

4 Results

4.1 Segmentation of beads

Existing code (`redseg.m`) for segmentation was modified to segment beads (`redsegbeads.m`) and keep only the suitable beads. The code follows the flowchart in figure (11). Edge detection and thresholding is used to identify beads. To remove beads that are on the edges, the function `regionprops` is called to calculate areas of the beads and area thresholding is carried out. Then, morphological dilation enlarges the beads, and beads that are too close would merge together to form one region, while beads that are sufficiently far apart would not merge (see figure (12)). The size of the dilation was decided based on visual inspection and testing how the mean intensity changes. Then areas can be calculated again and area thresholding is carried out to remove the unsuitable beads. The thresholding levels were decided by visual inspection and by studying the minimum and maximum areas of suitable beads after dilation (see figure (12)). The suitable beads are morphologically eroded with the same size and structural element to bring them to their original sizes. The output is labeled regions identifying suitable beads. The final output can be seen in figure (13). Next, the code calls functions to calculate the mean intensity and other properties which are saved in a .mat workspace which will later be used for the generation of heat-maps.

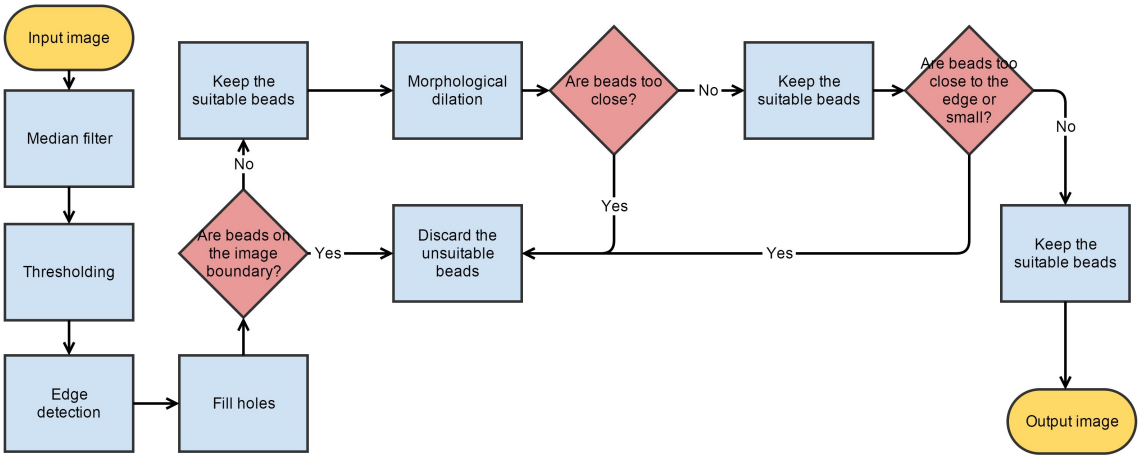
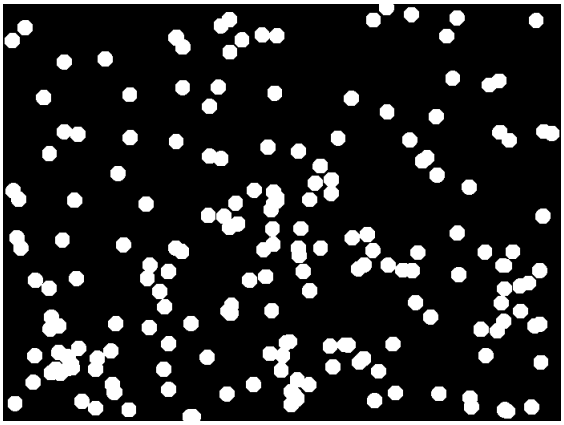
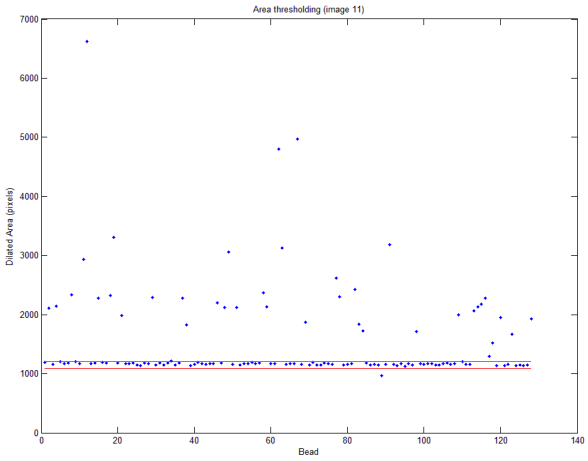


Figure 11: Flowchart to segment suitable beads



(a) Morphologically dilated beads show beads that are too close merging into a single area while beads that are sufficiently far apart do not.



(b) Area thresholding of dilated beads to remove merged beads (significantly larger areas) and small beads (significantly smaller area)

Figure 12: Key stages in excluding unsuitable beads.

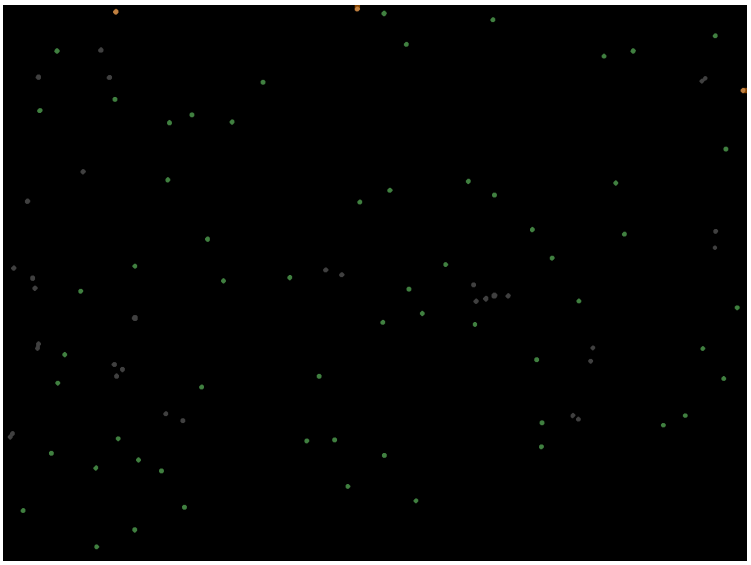


Figure 13: Red beads - too close to the edge or small. Grey beads - too close to each other. Green beads - suitable beads

4.2 Heat-map

The mean intensity in the field of view is best visualised as a heat-map generated with the script `Intensityheatmap.m`. It produces a heat map with the following steps:

1. Loading data from segmented beads (e.g. `Output.mat` the output of `redsegbeads.m`).
2. Calculating within a bin (`box_height` x `box_width`) the:
   (a) Mean intensities.
   (b) Number of beads.
   (c) Standard deviation of intensities.
3. Normalising the mean intensity.
4. Producing surface plots (viewed in 2D).
5. Optional scaling of calculated mean intensities matrix to full image size matrix for future use in correction.

The sizes of the bins were decided after testing (see figure (14)). Bins that are too large leads to loss of detail while bins that are too small can lead to empty bins (bins with no beads present). To account fo this trade-off, we chose the bin size of 52 x 58 pixels.

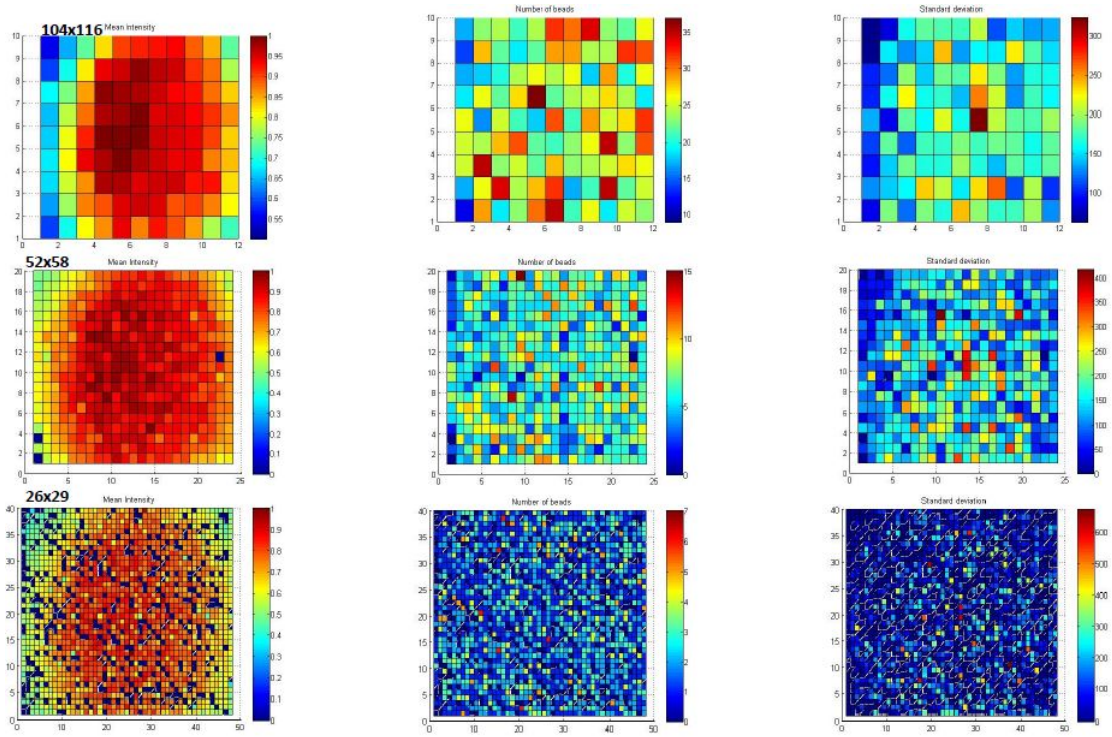
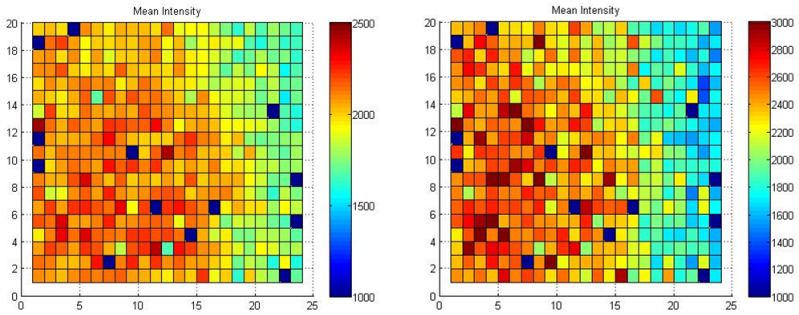


Figure 14: The mean intensities (left column), number of beads (middle column) and standard deviation of intensities for various bin sizes (right column). (The axes are in bin numbers with Images from beads1). As the bin size decreases, there are more empty bins (dark blue) but greater detail in the heat-map pattern.

The mean intensities were calculated for both RFP images and YFP[8] (yellow fluorescent protein) images, and they both illustrate the same intensity pattern as seen in figure (15). This pattern reveals that non-uniform illumination should be considered in the analysis.

[8] The yellow fluorescent protein is attached as a marker to indicate gene expression in bacteria



(a) The mean intensities of RFP (b) The mean intensities of YFP

Figure 15: The mean intensities of RFP (left) and YFP (right) images (The axes are in bin numbers)

4.3 Comparison

Code to segment fluorescent beads and generate a heat map was also previously developed by Om Patange. Hence, his code was used as a comparison to assess the process of segmentation and heat-map generation.

4.3.1 Segmentation

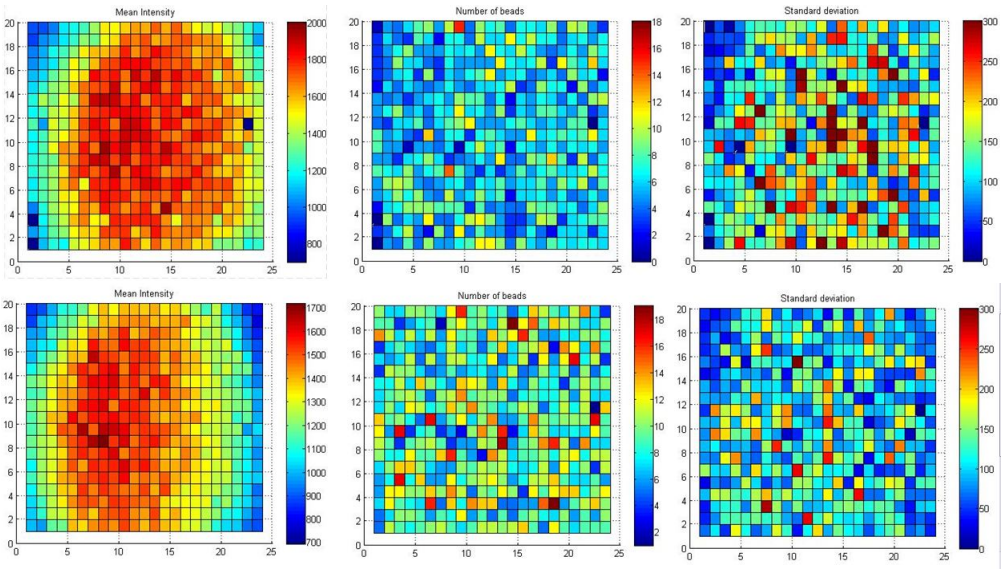


Figure 16: The mean intensities (left column), number of beads (middle column) and standard deviation of intensities (right column) for the bead segmentation code developed (above) and Patange's code (below). (The axes are in bin numbers).

Comparing the segmentation processes reveal that they both illustrate a similar fluorescence pattern in the field of view. Patange's code is less strict in removing beads that are close to each other, and hence retains more beads (as seen in figure (16) where the number of beads are higher). Moreover, there is background intensity subtraction, leading to lower mean intensities.

In the code that was developed, as many more beads are discarded due to its rigorous exclusion of beads that are close to each other, it would require a large number of fluorescent bead images to generate a more representitive heat-map.

4.3.2 Heat-map

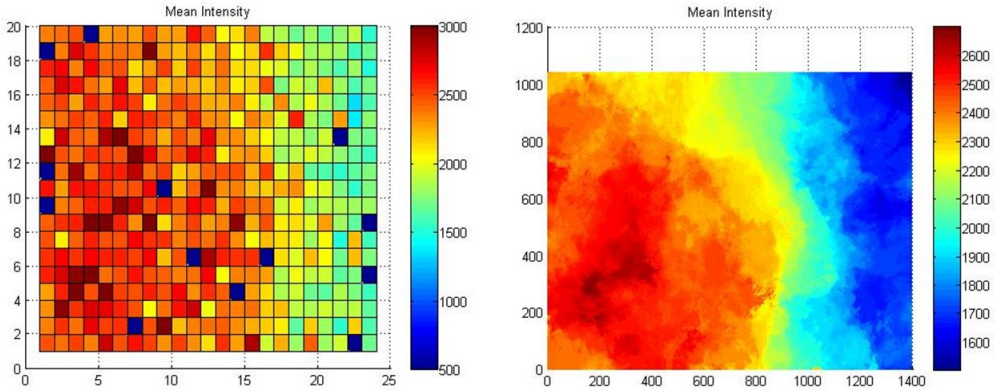


Figure 17: The heat-map of developed code (Left. The axes are in bin numbers) and of Patanage's code (Right. The axes are in pixels). (Images from beads1).

The heat-map generated with Patange's code carries out smoothing and averaging in a circular pattern. In comparison, the code developed here only averages the intensities within each rectangular bin. Hence, the heat-maps generated look slightly different. Patange's code is time consuming as it processes the heat-map pixel wise over the full image size. In comparison, the code developed processes the beads over the bins.

4.4 Segmentation method 1 - Watershed

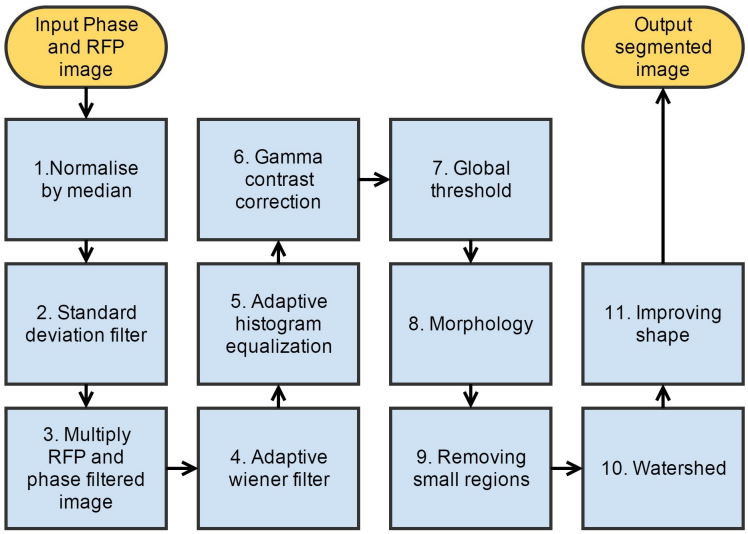
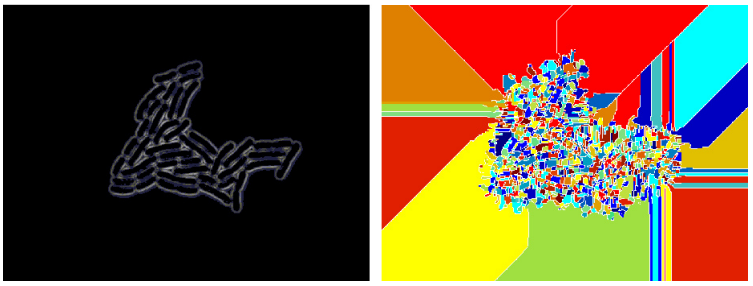


Figure 18: The Watershed method.

The Watershed method was developed with the fundamental principle that watershedding would always form complete cells. Moreover, it was developed to use both RFP and phase images which would provide more information for accurate cell detection. In addition, it works very differently from Schnitzcells in that there is no explicit edge detection but rather filtering and thresholding to get cell outlines. The method developed can be seen in the flowchart in figure (18). The majority of the algorithm is designed to *produce a suitable mask* that can be watershed.

4.4.1 Thresholding for a binary mask

Initially we normalise the frames by the median of the movie to make more uniform pixel intensities over the frames of the movie. Then steps 2 and 3 produce an edge enhanced image and Step 4 helps reduce noise (see section 3.6). Watershedding a grayscale image leads to over-segmentation as seen figure (19) where the watershed algorithm was applied to step 6. Applying the watershed algorithm to earlier steps would produce even more over-segmentation. Hence, thresholding and morphological operations are applied to produce a binary mask which improves results as seen in figure (19).



(a) Grayscale image after filtering and gamma correction

(b) Watershed result

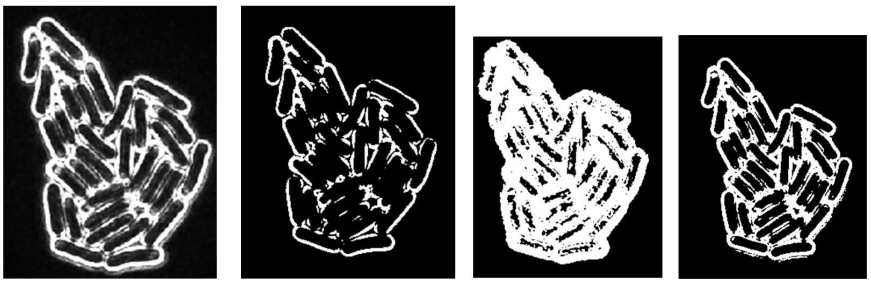
(c) Binary mask after thresholding

(d) Watershed result

Figure 19: The watershed result for a grayscale and binary mask. (Frame 161 Movie 3)

4.4.2 Gamma correction

For good global thresholding, gamma correction is vital as seen in figure (20). However, for the four different movies that were tested, the optimal gamma correction values were different. Rather than finding parameters to automatically adjust gamma correction levels, we aimed to identify a constant gamma correction for all movies. This lead to the development of step 5 - adaptive histogram equalization.



(a) The standard deviation filtered image (grayscale)

(b) Simple threshold of the image

(c) $\gamma = 0.3$ and threshold

(d) $\gamma = 0.4$ and threshold

Figure 20: Various gamma corrections and then thresholding on the standard deviation filtered image. We can see that a suitable gamma correction of $\gamma = 0.4$ and thresholding provides better results compared to a simple thresholding. (Image 140 Movie 4)

4.4.3 Adaptive histogram equalization

Different parameters were tested to enable a constant gamma correction for all movies (see section 3.5 for details on histogram adjustment). The best obtained result was with an exponential ($\alpha = 2$), bin size 8 with a constant gamma correction of 0.4. In tuning the Watershed method, Movies 2 and 3 produced suitable results. However, Movies 1 and 4 only produced suitable results after about frame 100 where the larger number of cells increases the distribution of brighter pixels (see figure (21)).

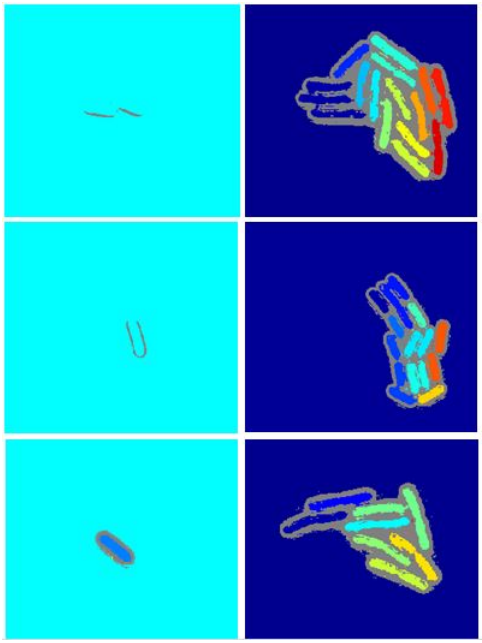


Figure 21: Preliminary results with the Watershed method. Top to bottom - Movie 1, Movie 4 and Movie 2. Left = Frame 1 Right = Frame 100. For movies 1 and 4, we can see incomplete cell shapes for the initial frames.

4.4.4 Results and challenges

Figure 22: Watershed method results

Results from the Watershed method can be seen in figure (22). There are complete cells however the method leads to the loss of precise cell shapes. Moreover, the performance is varied for different movies and can lead to merged cells which would be really difficult to fix manually. The reason for merged cells is due to incomplete cell edges when thresholding. *Any small connections lead to the overflow of the watershed basin.* Changing the morphological processes and gamma corrections to complete these edges would lead to over-segmentation. Hence there is a tradeoff - merged cells vs over-segmentation vs tuning for different movies. Results from Schnitzcells for comparison can be seen in figure (23).

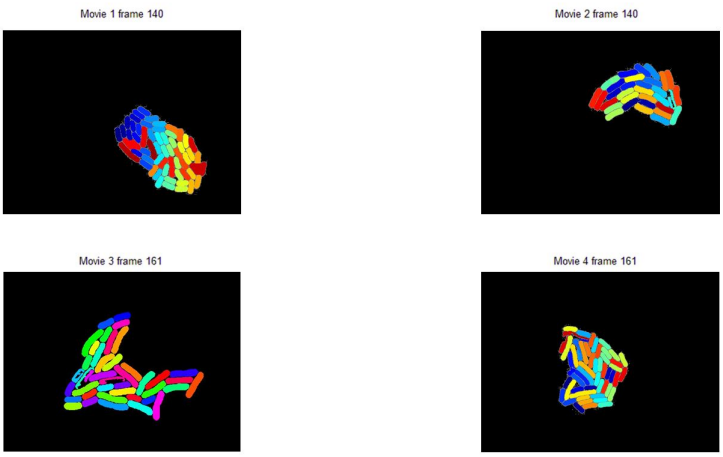


Figure 23: Schnitzcells method results. Movie 1 frame 140 shows merged cells. Movies 2, 3 and 4 show incomplete cells.

Improvements to the Watershed method can be made by exploring:

1. Adaptive thresholding to improve the masks (used in later methods).
2. Developing variable gamma correction to automatically change for each movie.
3. Normalising images before gamma correction (used in later methods to make processes more uniform for different movies).
4. Distance transforms to produce the mask for watershedding.

4.5 Segmentation method 2 - Marker watershed

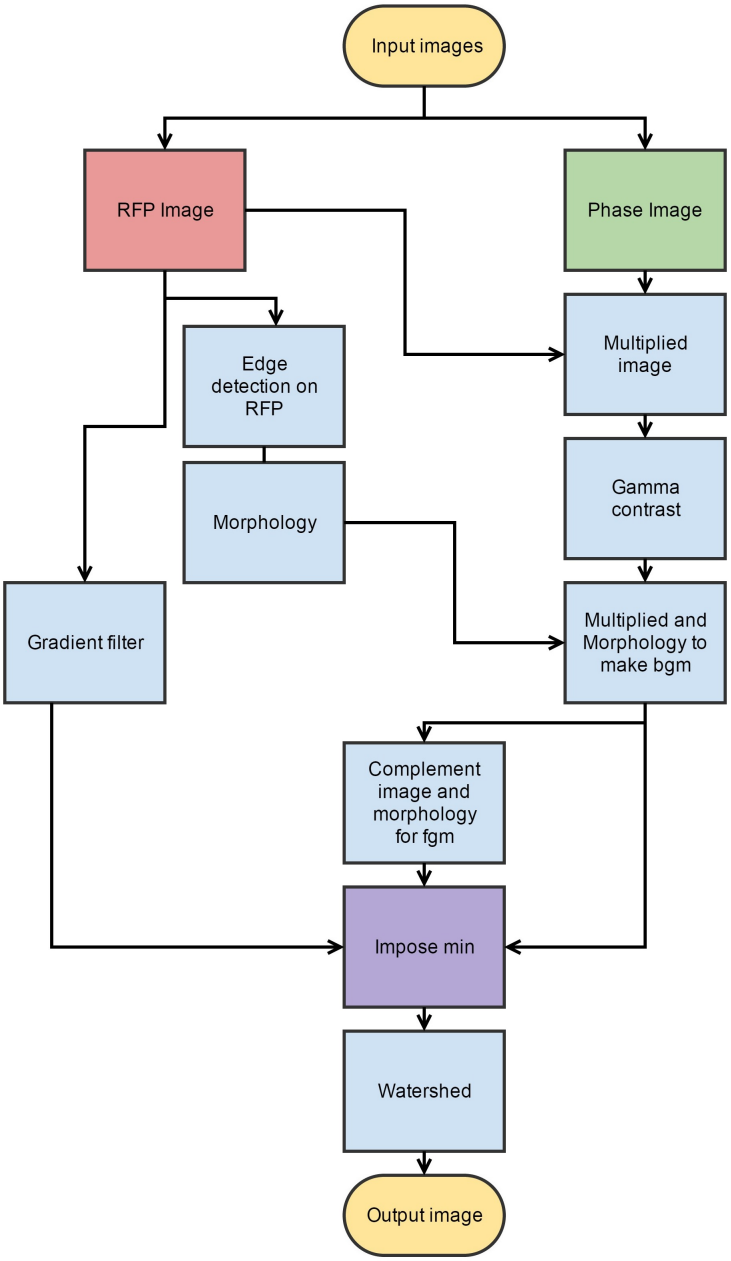


Figure 24: The Marker watershed method.

This builds on from Method 1 and is based on the technique followed by MATLAB Documentation on *Marker-Controlled Watershed Segmentation* [4]. This method aims to identify foreground (fgm) and background (bgm) markers which are then transformed to be the only minima in the image to be watershed. The steps for this method can be seen in figure (24).

4.5.1 Foreground marker

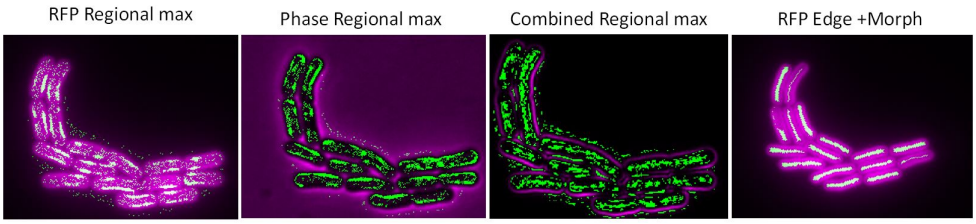


Figure 25: Various foreground markers processed from different images (using the function `imregionalmax`). Regional maxima method is very noisy and inconsistent. The best method was using edge detection and morphology as seen in the last image.

Foreground markers are used to mark the presence of a cell - ideally a single mark to mark a single cell. They are suggested to be found by identifying regional maxima. However, this proved to be very challenging to provide clear, consistent markers when tested with different movies, frames and parameters (see figure (25)). The best method identified was using edge detection and morphology on the RFP image.

4.5.2 Background marker

Background markers are used to mark everything that is not a cell - ideally all edges of the cells to maintain the cell shapes when watershed. They can be found by using distance transforms. However, this method did not encircle cells. The method developed here uses the combined (RFP x Phase) image, gamma correction, RFP edge detection and morphology. The markers are imposed as minima on the image gradient of the RFP channel image. The image gradient provides a suitable topological surface for the watershed function. (figure(26)).

(a) Foreground (pink) and background (green) markers.

(b) The Marker watershed result.

Figure 26: The Marker watershed method.

4.5.3 Results and challenges

It was vital to ensure that the different markers did not overlap. Hence, foreground markers were processed by using the complement of background markers. Identifying good consistent markers remains a challenge as edge detection produces edges within cells as well. The final results seemed somewhat promising - slightly better cell shapes than Method 1. Moreover, this method could be developed to mark cells from previous frames to watershed. This would enable tracking to be developed with segmentation. *Good marking can lead to good results* seen in figure (26). However, due to time constraints in tuning this method to identify good markers, it was not developed further. For comparison, images from the marker watershed method can be seen in figure (27).

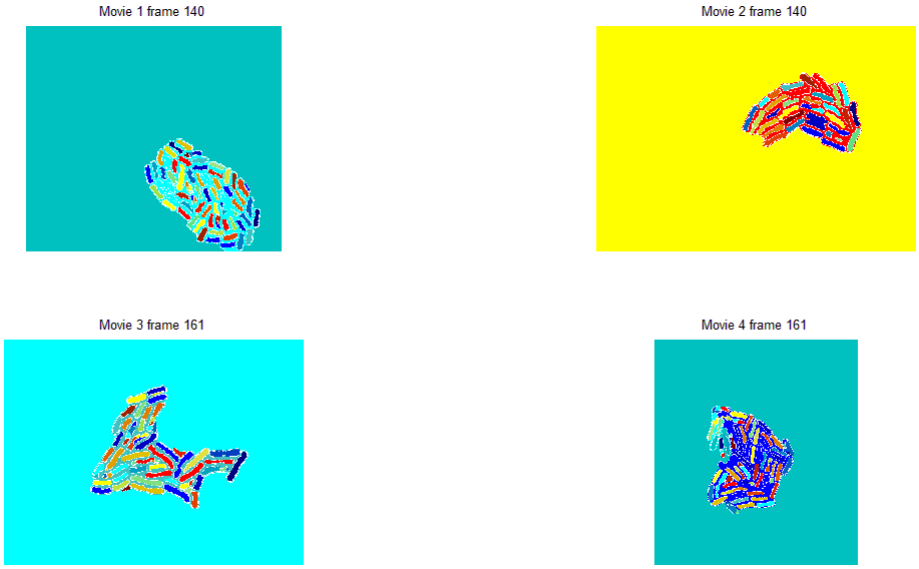


Figure 27: Marker watershed method results

4.6 Segmentation method 3 - Gamma correction and threshold

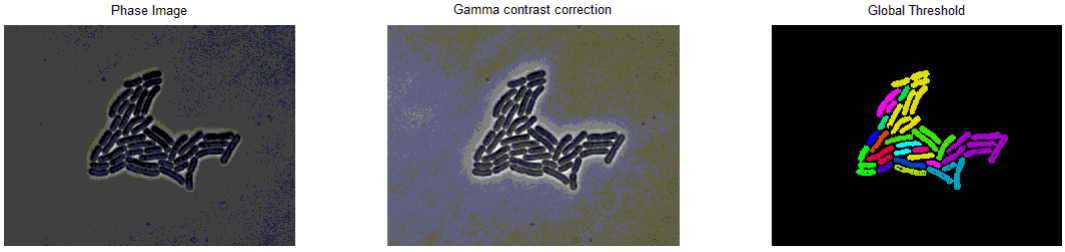


Figure 28: Gamma correction and Thresholding method.

This method is applies image normalisation, suitable adaptive filtering, gamma contrast adjustment to phase images and then globally thresholding for segmentation.

4.6.1 Results and challenges

This simple strategy is able to acquire good cell shapes and segmentation but produces artefacts of connections between cells. This is due to the close proximity of cells and cells touching each other in the microcolonies. Moreover, different movies require different gamma correction for good thresholding. In addition, the phase images can be unreliable due to variations in the lighting in the experimental set up. *Normalising the phase image before gamma correction enables more uniform gamma correction performance over different movies.* When testing with an adaptive threshold, the segmentation improved slightly.

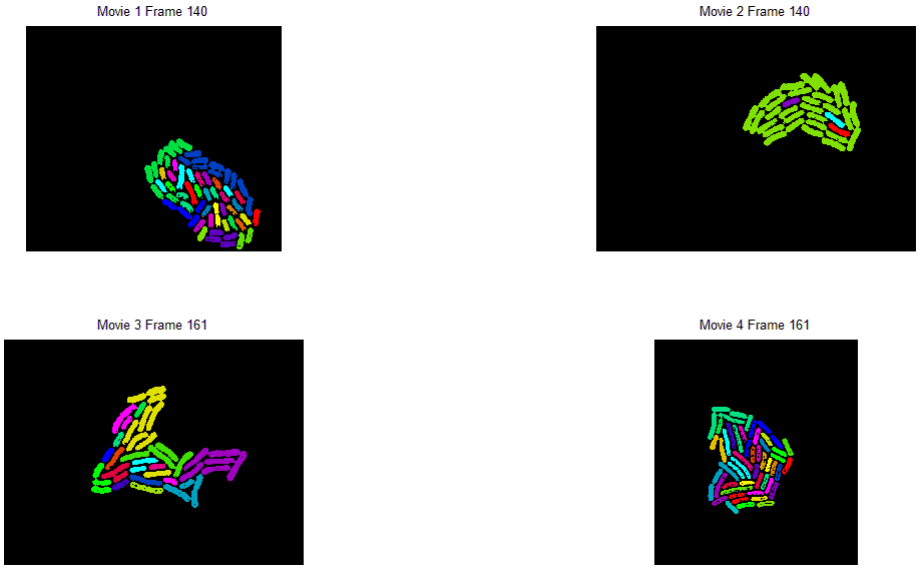


Figure 29: Gamma correction and Thresholding method results

4.7 Segmentation method 4 - Hybrid method

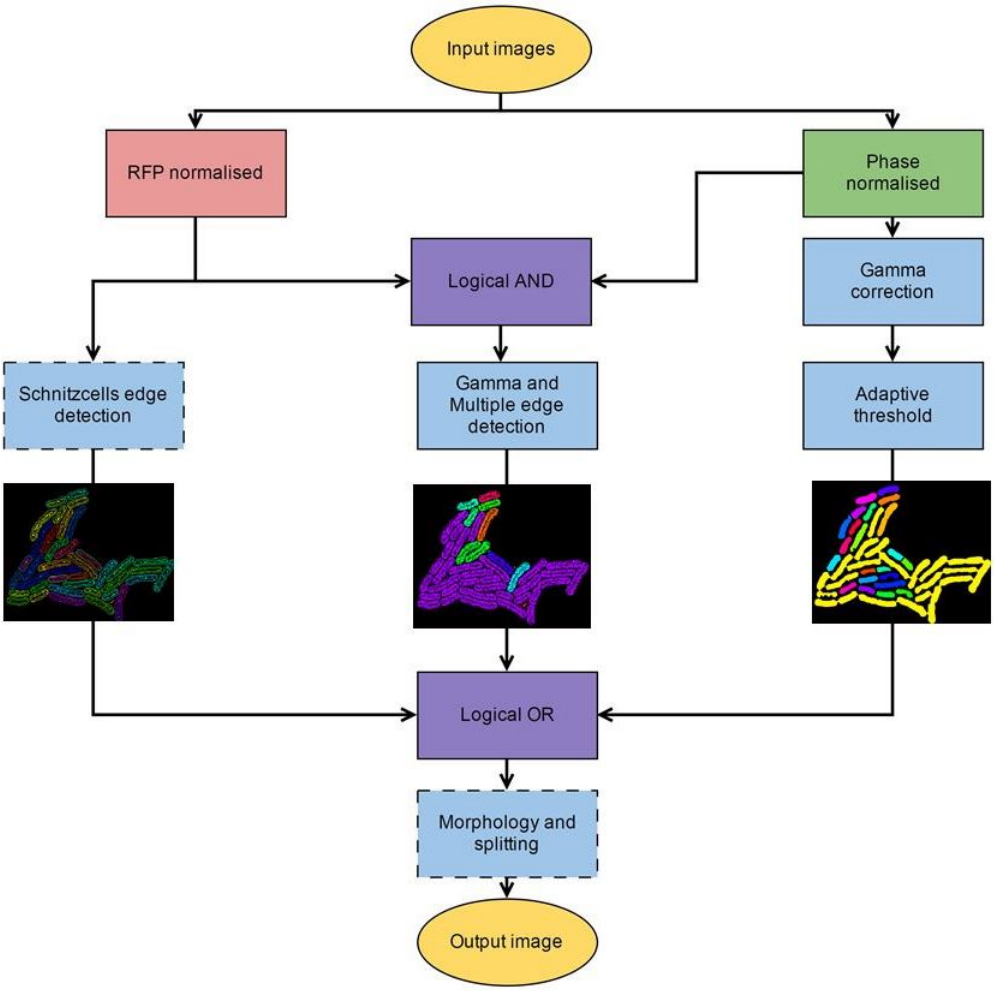


Figure 30: The Hybrid method.

The hybrid method aims to use the strengths of different image segmentation methods such as edge detection and thresholding to produce better results. The method developed can be seen in figure (30).

4.7.1 Schnitzcells

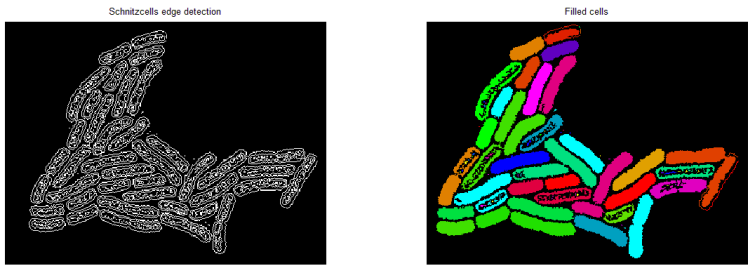


Figure 31: (Left) Edge detection. (Right) Filled shapes using *imfill* from the detected edges.

The edge detection used in Schnitzcells (Laplacian of a Gaussian method with the edge function) *performs very well in detecting the outlines of cells from RFP images.* However, the edges within cells due to the non-uniform fluorescence within a cell lead to difficulties in the proceeding step of filling closed shapes. The shapes are filled with the function `imfill` which can fill 'holes' with 4-pixel connectivity. Any

erroneous edges or incomplete edges lead to incomplete or merged shapes at this step as seen in figure (31).

Testing the `imfill` function with different parameters such as filling 'holes' with 8-pixel connectivity together with morphological operations to fill shapes (`bwareaopen` and `bwmorph`) to improve the filling of cells was difficult to optimize for different movies. There always proved to be a trade-off between too many edges and the fill forming 'blobs' or too few edges and the fill forming incomplete cells.

Combining RFP and phase images would improve edge detection for the `imfill` function and this is what is used in the hybrid method.

4.7.2 Combined image

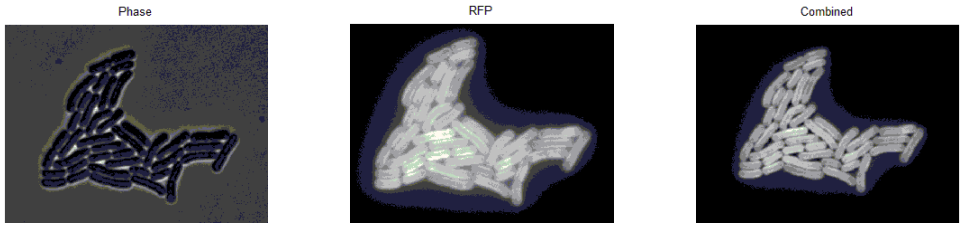


Figure 32: Combining images.

One of the key stages in the hybrid method is to combine the normalised phase and RFP images to create an image that highlights the edges between cells more clearly. To do this, the inverse of the phase image is multiplied to the RFP image. Hence, regions in the RFP image where the fluorescence bleed leads to higher intensities, the multiplication of a phase image where the edge is clearer would reduce this intensity. This is similar to the function of a logical AND - only regions where RFP and inverse phase have high intensities do the output image have a high intensity.

The combined image is gamma corrected and then used for edge detection at various sensitivities (standard deviations of the edge filter). The aim of this procedure is to produce multiple edges which to an extent, 'fill' each cell. In addition, edge detection at the same sensitivity of Schnitzcells (standard deviation = 0.9) is used with `imfill`, and produces improvement in filled shapes (though not complete cell detection).

4.7.3 Phase image

The key advantage of the phase image is producing filled cell shapes using Method 3 (Gamma correction and thresholding). Hence performing adaptive filtering, gamma correction and adaptive thresholding of the phase image produces filled cells and cell shapes, but with interconnections. To reduce these connections, the image is then morphologically opened and existing code from Schnitzcells is used to break big cells (see section 4.7.5 below).

4.7.4 Hybrid step

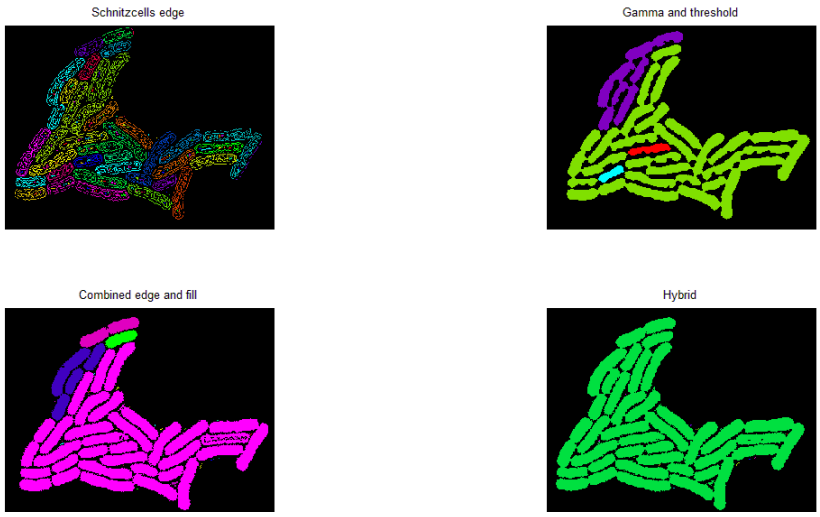


Figure 33: Stages in producing the hybrid image.

The images generated from:

1. Schnitzcells edge detection (good cell outline).
2. Combined image's multiple edge detection and fill (good cell outline and fill).
3. Thresholded phase image (good cell fill).

are added together to create the hybrid image. Here all cells will be detected - regions where thresholding or edge detection were poor are corrected for by each other. This is similar to the logical OR operation, where cell detection from either edge detection or thresholding would produce an output. There is a lot of redundancy at this stage, but proves to be advantageous as it ensures cell detection in different movies on different frames. These steps can be seen in figure (33). The output is always over-connected, with some occasional noise and small holes within cells.

4.7.5 Morphology and splitting

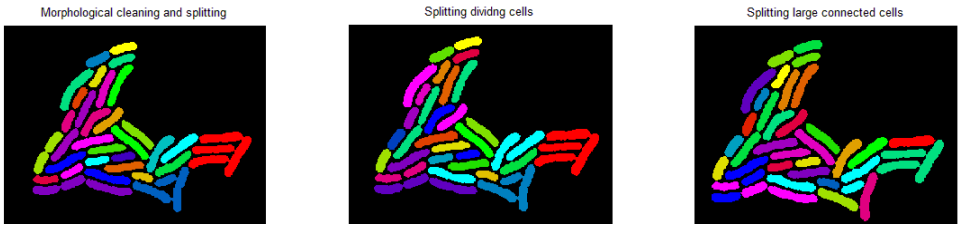


Figure 34: Stages in morphology and splitting performed on the Hybrid image in figure (33) segments the cells.

A series of morphological operations are used to remove the over-connected cells, noise and fill the small holes within cells. These steps include filling isolated pixels, morphological opening, morphological erosion and breaking small connections pixel wise (`bwmorph`). In addition, existing code from Schnitzcells is used to further segment the cells and split large dividing cells. This is performed in two steps :

1. Cutting individual cells at the narrow waist - splits cells where both sides of the cell are sufficiently concave.
2. Breaking big cells - This is performed by going through the cells and looking for a change in the phase value which would indicate a space between cells.

These stages can be seen in figure (34) to finally produce good segmented cells.

4.7.6 Results and challenges

In testing the four movies, this new method produces better segmentation than Schnitzcells. Playing on the strengths of the different methods improved the overall performance (see figure (35)).

The segmentation can occasionally have errors of merged cells when the phase image is poor. In future, this could be improved by weighting the images at the hybrid step with a lower weighting for the thresholded phase image.

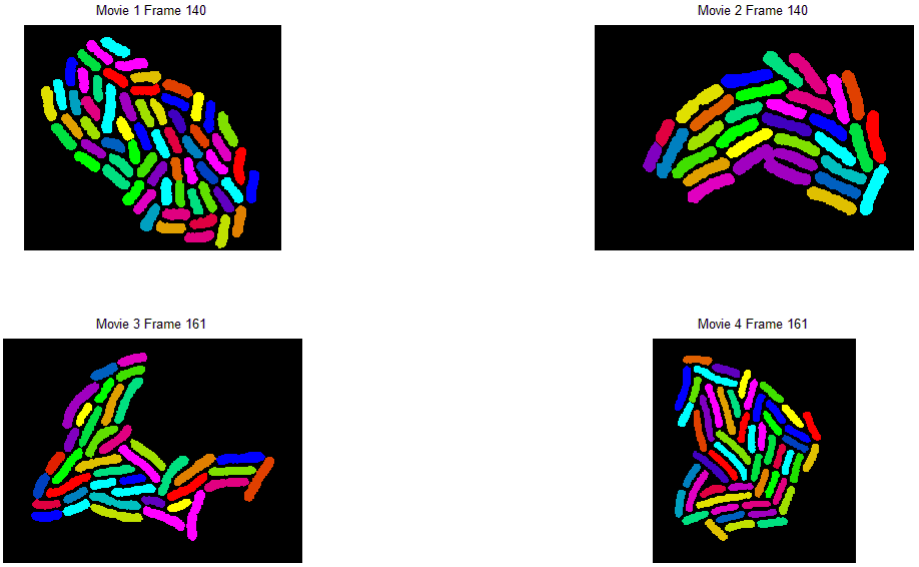


Figure 35: Hybrid method results. We see much better segmentation compared to Schnitzcells. In Movie 2 frame 140, we see a little merging of cells.

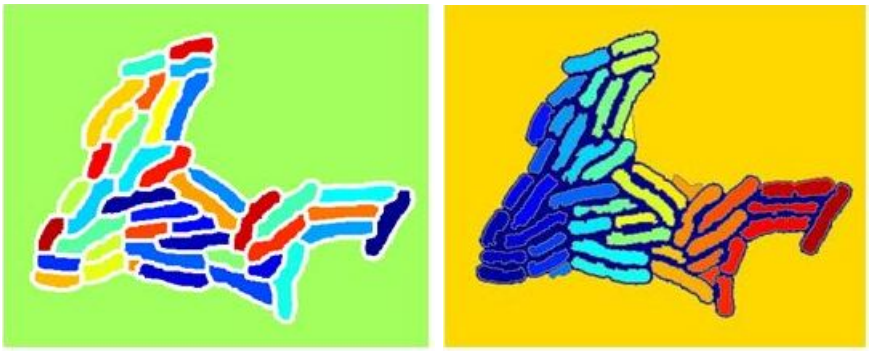
5 Conclusions

The different investigated methods have their advantages and disadvantages as seen in table (1).

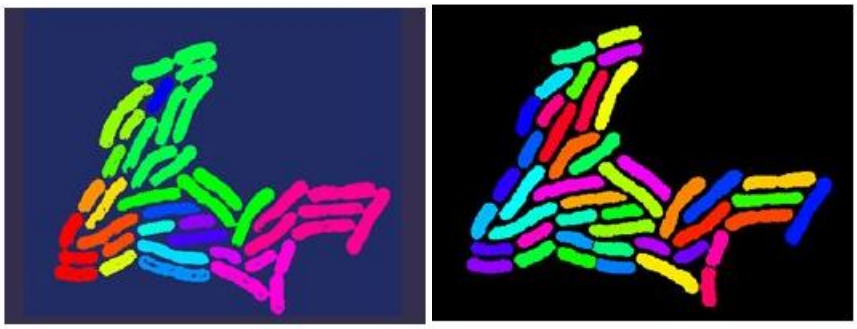
Comparing different segmentation methods			
Method	Advantages	Disadvantages	To improve in future
Method 1 - Watershed	Filled cells	1. Requires tuning of thresholding to work on different movies. 2. Loss of precise cell shape 3. Overflow of catchment basin forms merged cells	1. Adaptive thresholding 2. Automate gamma correction level/ normalise images 3. Distance transforms to produce the mask
Method 2 - Marker Watershed	1. Does not require fully closed edges 2. Can be developed for tracking	Variable results and shape depending on quality of markers	1. Improve bgm and fgm 2. Improve mask applied
Method 3 - Gamma contrast	Good shape detection	Requires Gamma tuning for different movies	1. Automate gamma correction level 2. Add segmentation code to chop merged cells
Method 4 - Hybrid	Very good segmentation	Occasional merged cells	Improve the use of phase threshold image

Table 1: Summarising the different segmentation methods

With suitable tuning, the methods produce results seen in figure (36). The best method is a hybrid method of edge detection, gamma correction, and adaptive thresholding. The results show an improvement over the current software. Future work would look into incorporating information from previous frames to improve tracking and machine learning.



(a) Watershed Method (b) Marker watershed result



(c) Gamma threshold result (d) Hybrid result

Figure 36: The results of the four segmentation methods investigated. For comparison, the Schnitzcells result can be seen in figure (4).

References

[1] Jonathan W Young, James C W Locke, Alphan Altinok, Nitzan Rosenfeld, Tigran Bacarian, Peter S Swain, Eric Mjolsness & Michael B Elowitz, *Measuring single-cell gene expression dynamics in bacteria using fluorescence time-lapse microscopy,* Nature Protocols 7, 80–88 (2012)

[2] Guanglei Xiong, *MATLAB Central File Exchange*, Tsinghua University, (2006) http://uk.mathworks.com/matlabcentral/fileexchange/8647-local-adaptive-thresholding

[3] Math Works, *Image Processing Toolbox User's Guide,* The Math Works, Inc, (2015)

[4] Math Works, *Marker-Controlled Watershed Segmentation,* The Math Works, Inc, (2015) http://uk.mathworks.com/help/images/examples/marker-controlled-watershed-segmentation.html

[5] N. Otsu, *A Threshold Selection Method from Gray-Level Histograms*, IEEE Transactions on Systems, Man, and Cybernetics, vol. 9, no. 1, pp. 62-66, 1979.

[6] Math Works, *The Watershed Transform: Strategies for Image Segmentation,* The Math Works, Inc, (2015) http://uk.mathworks.com/company/newsletters/articles/the-watershed-transform-strategies-for-image-segmentation.html

6 Appendix

6.1 Fluorescence in field of view code

Algorithm 1 redsegbeads.m for Bead Segmentation

```
% Modified version of REDSEG to segment and keep suitable beads which can be used to produce a heatmap
% Outputs eg 'Sbeads_001-1.mat' which stores the results of segmentation.
% This will be the input to INTENSITYHEATMAP.M
% Call REDSEGBEADS within the folder with RENAMED IMAGES eg 'Blah_001-01-p.tif'
% Example:
%
% redsegbeads('naturalback',1);
%
% The subfunction PROGTHRESHFLOUR carries out the bead segmentation
% PROGTHRESHFLOUR follows the steps described in section 4.1
```

Algorithm 2 Intensityheatmap.m for Heat-map generation

```
% Used to produce the distribution of fluorescence in the field of view
% Example :
%
% Intensityheatmap
% When user input is requested 'What is the filename? '
% Key in eg Sbeads_001-1
%
% Produces a heat map with the following steps:
% 1. Loading data from segmented beads (output file of REDSEGBEADS eg % Sbeads_001-1.mat)
% 2. Calculating
% (i)mean intensities
% (ii) number of beads
% (iii) standard deviation
% within a box (box_height x box_width)
% 3. Normalising the mean intensity
% 4. Producing surf plots (viewed in 2D)
% 5. Optional scaling of calculated mean intensities matrix to full image size matrix for future use
```

6.2 Hybrid segmentation code

Algorithm 3 segmoviefluorH.m used to run the hybrid method within Schnitzcells for a time-lapse movie.

```
% Modified version of SEGMOVIEFLUORC to incorporate the HYBRID method for segmentation.
% Example :
%
% p = segfluormovieH(p);
%
% It is modified to read RFP images into the variable X and phase images into the variable Y.
% It is modified to call SEGFLUORH(X,Y,p)
```

Algorithm 4 segfluorH.m is used to run the hybrid method within Schnitzcells for a single frame.

```
% Modified version of SEGFLUORC to incorporate the HYBRID method for segmentation.
% Example :
%
% L = segfluorC(Im_phase,Im_RFP,p);
%
% 1. Filter (segfluorC) - to output image col_center for Mask
% 2. Mask (segfluorC) - to output rect
% 3. Hybrid method of segmentation - calls HYBRIDSEG(PHASEIM,PH3) to output L9 the labeled cells
```

Algorithm 5 hybridseg.m is used to run the hybrid method on a single frame.

```
% Carries out the hybrid method of segmentation developed.
% Example :
%
% L = hybridseg(PhaseIm,RFPIm);
%
% Where L is the labelled region of cells
% PhaseIm = The phase contrast image
% RFPIm = The RFP channel image
%
% Hybrid method of segmentation carries out :
%
% 1. Normalising and combining images
% 2. Gamma contrast correction on combined image
% 3. Phase image gamma correction and adaptive threshold
% 4. Phase image breaking big cells
% 5. Multiple edge detection and fill
% 6. Schnitzcells edge detection
% 7. Combining multiple edges+phase thresold + fill
% 8. Morphology for splitting cells
% 9. Cutting cells at narrow waist
% 10. Breaking big cells
```