Computer Vision Methods in Biological Cell Tracking and Cell Motion Analysis

Name: Qianyi Li
zID : z5343150

Name: Ziqiao Lin
zID: z5324329

Name: Yuchao Fang
zID: z5155149

Name: Sihang Wang
zID: z5223840

Name: Fei Yu
zID:z5174228

1 Introduction

Biological cell tracking is an important computer vision task in cell biology to analyse the behaviour of the cells. Cell tracking is a subsection of multi-object tracking and requires the analysis of the behaviours of the cell, including displacement, split, appearance, and disappearance.

The input of the task is a sequence of time-lapse 16-bit images of cells taken by a microscope. The images are continuous and in the order of time. However, since the pixel value of the original image has a small range, the input image requires preprocessing before the segmentation. For better visual effect and more straightforward calculation, the preprocess will transform the images into 8-bit of bit depth and use 0-256 as the pixel value range for the following process. The preprocessed images will then enter the workflow of the task.

fig 1.1 - the example of an input image

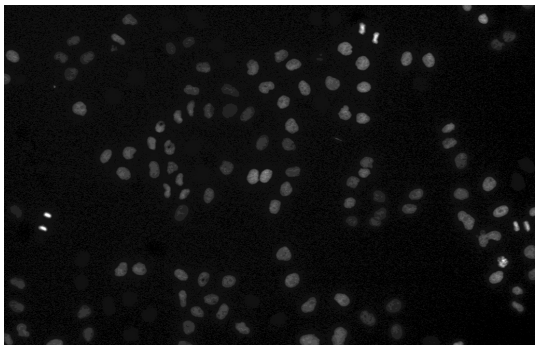

fig 1.2 - the example of a preprocessed image

The biological cell tracking task includes two steps, segmentation and tracking. The segmentation step would process every sequence image individually, identifying, extracting and labelling every cell. In this step, the demo will read the sequence of images separately, and every image will be segmented, and each cell will be labelled identically (fig 1.3). The sequence of segmented images will be the input of the tracking step.

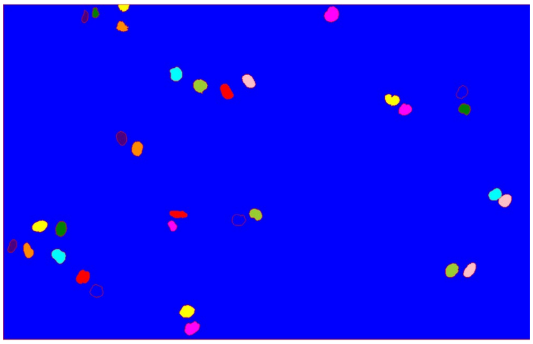

fig 1.3 - the example of a segmented image

The tracking step would use the sequence of the segmented images to show the movement. This step will identify the same cells in the whole sequence of images based on the shape and the location of the cells. This process will draw the track of the movement of each cell.

Apart from tracking the cells, the demo also includes some analysis of the cell movement.

The cell movement consists of the cell count, the average cell area, travel distance, and marking the splitting cell. The output image will also include such information (fig 1.4).

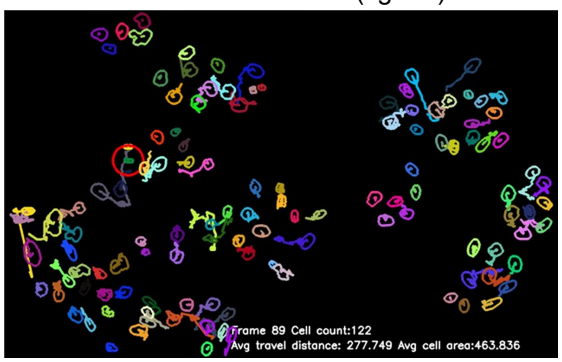

fig 1.4 example of output

The demo has referenced a few pieces of literature and compared different methods. This report will mention the literature review, the methods comparison, experimental results of the demo and some discussion about the problems in the experiment process.

2 Literature Review

Cell tracking methods usually have two major steps: cell segmentation and cell association. Segmentation will turn an image into a new one with different parts that can be recognised and meaningful. A common way to achieve this is by first obtaining a binary image which has all of the cells marked in white pixel with background being dark. However, such method will usually result in the output image containing too much noise since usually most of the binary images are obtained simply by applying a simple image pixel threshold method [1]. In this project, to effectively remove the noise and detect the cells which are displayed in different grey pixel values, a more sophisticated binary image-acquiring module should be developed. After the input image is successfully turned into a satisfying binary image with cells clearly labeled in white pixel, the watershed segmentation can be applied to the binary image with distance transform as the potential markers.

The next step is to associate cells over time. Kalman filter has shown good performance on time-lapse images in cell tracking [2]. Matching similar cell pairs by their positions on the local graph for any two consecutive time points was used too. Kalman filter-based matching has been proven as an effective and efficient method on cell tracking. However, Kalman filter-based matching module does not really provide a detailed and easily manageable prediction process. For the simplicity and more customizable structure, a property-weighted algorithm should be more flexible. Thus, after reviewing some options, it is reasonable to decide the final module structure based on the actual performance and implementation difficulty of this project.

3 Methods

3.1 Preprocess

3.1.1 Normalization

Observing the images in the dataset shows that it is difficult to find that the original images (fig 1.1) are in background coloured grey and cell pattern coloured light grey, which makes the patterns very difficult to recognise. Therefore, we need preprocessing first in order to distinguish the background and patterns easily.

After reading the grayscale image, the first thing is normalisation to adjust the background pixels to 0 while ensuring that the highest and lowest pixel differences remain unchanged. As in the cell image, the pixel values distribution is relatively concentrated, and the minimum and maximum values are stable. Therefore, min-max normalisation is chosen as normalisation to get the following result (fig 3.1.1).

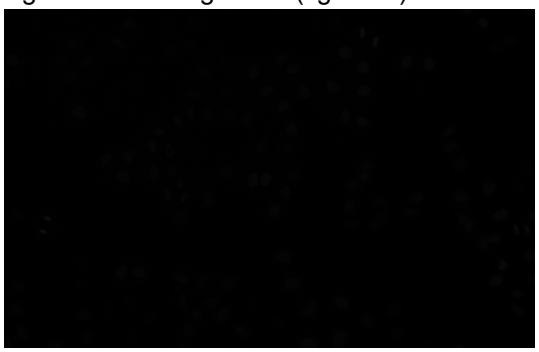

fig 3.1.1 - the normalised image

For the normalised image (fig 3.1.1), it is difficult to recognise the cells only with eyes

even after thresholding because of the slight pixel value difference. In order to make a better visual effect, contrast stretching will be applied (fig 3.1.2).

Note: Applying contrast stretching is only for good visual effect in the report. When doing experiments, contrast stretching is not applied in order to avoid errors.

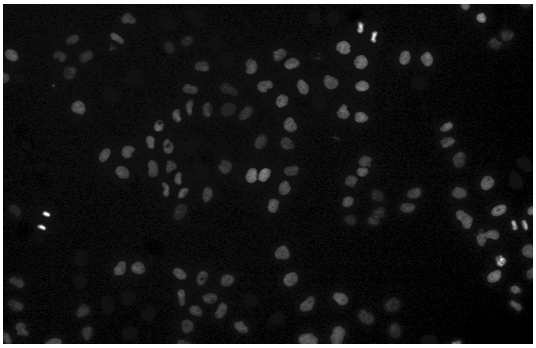

fig 3.1.2 - the normalized image (streched)

3.1.2 Thresholding

The watershed algorithm in the segmentation process will require another further preprocess, thresholding. Since the pixels of each cell are not the same, in order to highlight the cells, adaptive thresholding can be used instead of simple thresholding. Unlike the threshold value is global in simple thresholding, the adaptive threshold will calculate the threshold value for smaller regions in order to get better results [3]. The resulting image after the adaptive threshold is attached below (fig 3.1.3). Similar to normalised, a contrast stretched version is attached below for better visual effect (fig 3.1.4).

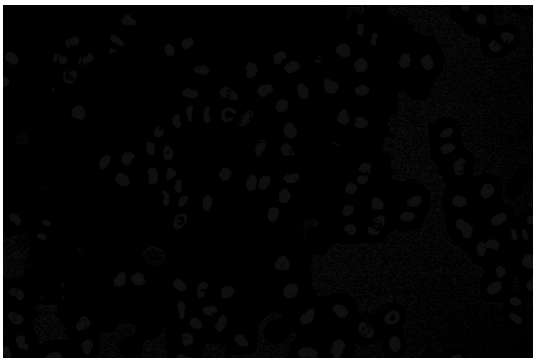

fig 3.1.3 - The image after adaptive thresholding

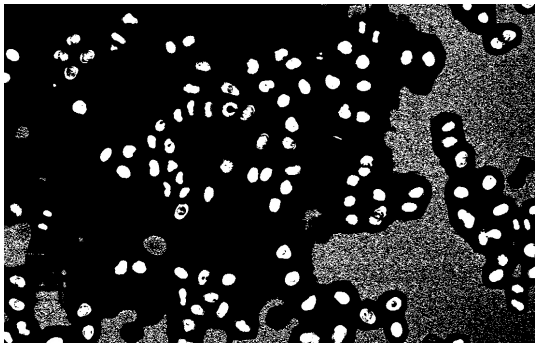

fig 3.1.4 - the image after thresholding (Stretched)

For the image after thresholding (fig 3.1.4), it is clear that there are many noises on the right half of the picture. In order to remove these noises, erode and dilate are used. Eroding all the patterns in the picture can effectively remove the noises [4]. Dilating the picture can effectively restore the corroded part [5]. Theoretically, the order of eroding and dilating can be changed, but problems occurred in experiments, which will be described in the discussion section later. Using morphologyEx in cv2 (erode first and then dilate) would obtain the following results after stretching.

A clear image (fig 3.1.5) will be ready for segmentation after the preprocessing methods mentioned above. After preprocessing is the segmentation part.

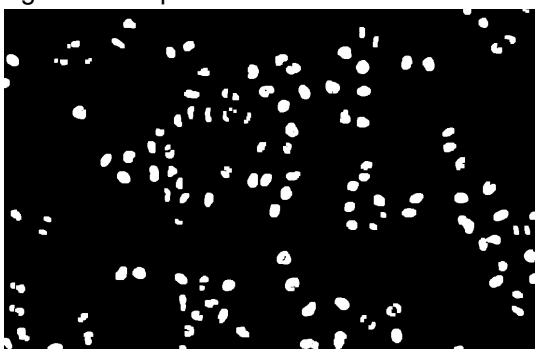

fig 3.1.5 - The image noises removed (Stretched)

3.2 Segmentation

Identifying and labelling every single cell identically in every image is the goal of the segmentation process. Drawing contours, the Mean Shift method, and the Watershed algorithm are the potential methods to achieve this goal. In the experiment, the mean shift did not perform well and also was very slow in time.

Therefore, this part will not discuss the mean shift method.

3.2.1 Watershed

In biological cell tracking tasks, it is common to see cells overlap with each other. Dealing with overlapped cells is a challenge in the segmentation process and essential for the tracking process afterwards. The Watershed method requires another preprocessing, thresholding (mentioned in 3.1.2) and contains three steps, calculating the distance, marking the markers and labelling.

The thresholding process in the preprocessing will provide thresholded images whose pixel values of the background pixels will be 0, and the values of object pixels will be 255. Using the thresholded image to calculate a distance image is the first step. When the transformation is complete, the pixel value of the distance image (fig 3.2.2) will be the distance from the current pixel to the nearest background pixel on the thresholded image. The pixels with higher pixel values will represent lower "altitude". When the "flood" comes in, they will go "underwater", and the algorithm will identify them as objects.

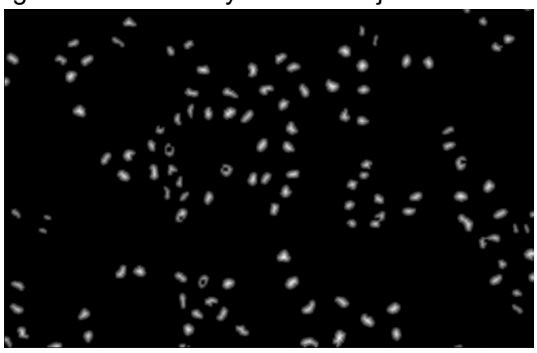

fig 3.2.2 - The distance image

Generating a marker image will be the second step. The marker image shows where the "flood" comes in. Thus, there should be as many markers on the marker image as the amount in the original cell. The program will assign unique labels to each marker, which will be the label of the cell. For this task, the program will erode the distance transform image for only a few iterations so that the resulting image will keep all of the cells of reasonable sizes. Furthermore, the resulting final markers are generated as below (fig 3.2.3).

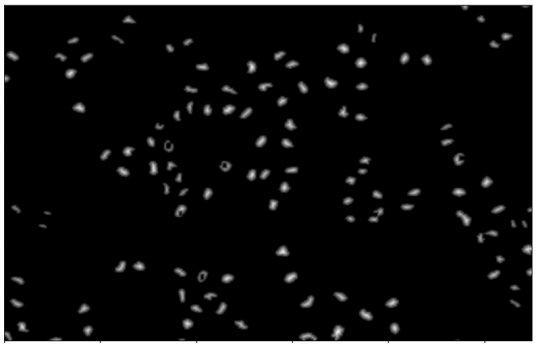

fig 3.2.3 - The marker image

The last step will be performing watershed on the image. The "flood" will come in where the markers are and fill the "lower" area (the pixels with the higher pixel values). The flood will fill every cell on the image with an identical label, which the algorithm will assign to the cell. When this step is complete, each cell will be segmented and labelled identically (fig 3.2.4).

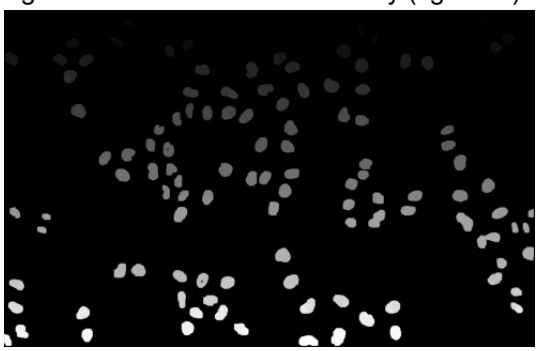

fig 3.2.4 - the result image of the watershed process

After comparison, this method is an ideal one for segmentation.

3.2.2 Find Contour

Finding contour is also another method for segmentation. This method segments the image by identifying the edges of the objects. The experiment has compared two ways of finding contour. The first one is directly performing the algorithm on the preprocessed image. However, the resulting image (fig 3.2.5) is not optimised as it contains noticeable noises and cannot be the input for tracking.

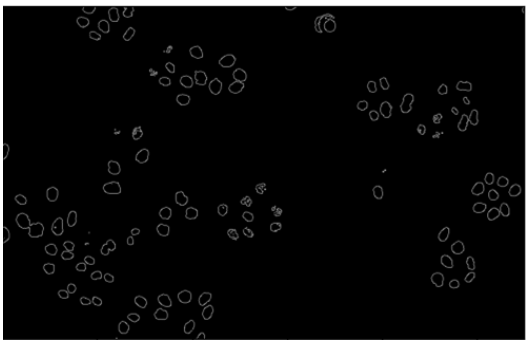

fig 3.2.5 - contours without using watershed

The other way is to perform the algorithm on the resulting image of the watershed method. This result (fig 3.2.6) is the more satisfying one among the other results.

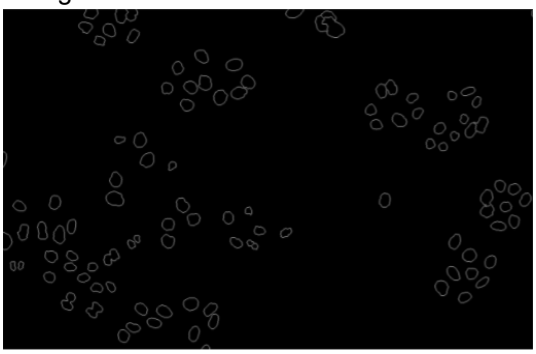

fig 3.2.6 - the contour of the watersheded image

It will require relabelling the cells to obtain the final segmented image, which is attached below. For better visual effect, the final result is printed with colour (fig 3.2.7).

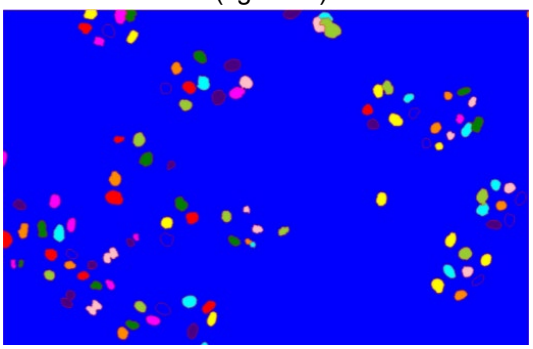

fig 3.2.7 - coloured segmented image

3.3 Tracking

The tracking module is based on connecting cells from the previous frame to their next frame [6]. The tracking process uses the Hungarian Algorithm to match the cells in the continuous frames. Distance and shapes are the main features of the cells during cell matching. Since the tracking process would require two continuous images, the following part will mention two continuous images. For a better description, the earlier image will refer to the image with an earlier time. The later image will refer to the image taken one frame after the earlier image.

The two features for deciding whether the two cells in the earlier and later image are the same cell are distance and shape. Distance is the first feature to consider. The pixel in the centre of the cells in the images will represent the whole cell in the distance calculation. A distance matrix is calculated for cell matching. The matrix is based on the Euclidean distance between the cells in the earlier image and the later image. The values in the distance matrix are in the range of [0,1], where smaller values mean smaller distance and a higher possibility to match these two cells in the earlier and later images.

Similar to the distance feature, another matrix will be calculated based on the shape of the cells. The shape matrix also consists of values in the range of [0,1]. This matrix shows the similarity between the cells in the two images. If the two cells are similar in shape, they would be more likely to be the same cell, thus having a smaller value in the matrix.

After completing the distance and shape matrix calculation, a cost matrix will be calculated as a linear combination of the distance and shape matrix. The cost matrix also has a value range of [0,1], where a smaller value represents a lower cost for matching the two cells. The Hungarian algorithm will minimise the overall cost for matching. Thus, the Hungarian algorithm will match the cells with the lowest overall cost, best matching the cells in the two images as bipartite [7].

There are many different cases to discuss based on the result of matching [8]. Suppose a cell in the later image has successfully matched with a cell in the earlier image. In that case, it will inherit the cell ID from that cell if they are the predecessor-successor relationship. However, if a cell has not matched in the earlier image, it could be a newly appeared cell. Alternatively, it could be a cell that just entered the microscope range, under segmentation, overlapping or a cell split from the parent cell. For newly appeared cells they will be assigned

a new cell ID and gets tracked. Besides, if a cell in the earlier image has not matched any cell in the later image, it will be labelled as disappeared. Record the information of this cell as its ID may be used again. If the cell shows up again, we can reuse the ID by comparing the information of the disappeared cell. If the new cell fits the condition of dividing according to (topological features), we assign a new ID for both another child cell and this cell.

After matching every continuous image, the tracking part is complete. An example image with cells labelled with their cell IDs is attached below (fig 3.3.1)

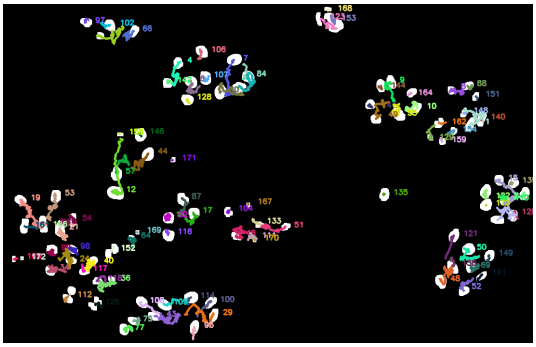

fig 3.3.1 - cells with IDs labelled by tracking

3.4 Cell Analysis

3.4.1 Cell Count and Area

By finalising the cell segmentation method, the foundation is already built for the cell amount analysis and cell average area calculation.

The cell amount analysis is aimed at the successful detection of all cells on the graph. For mathematical reasons, the cell average area calculation will exclude those cells lying on the boundaries of the cell image. This will require two versions of the cell segmentation result image so that both tasks can be done, and such different versions of the result are present below (fig 3.4.1)

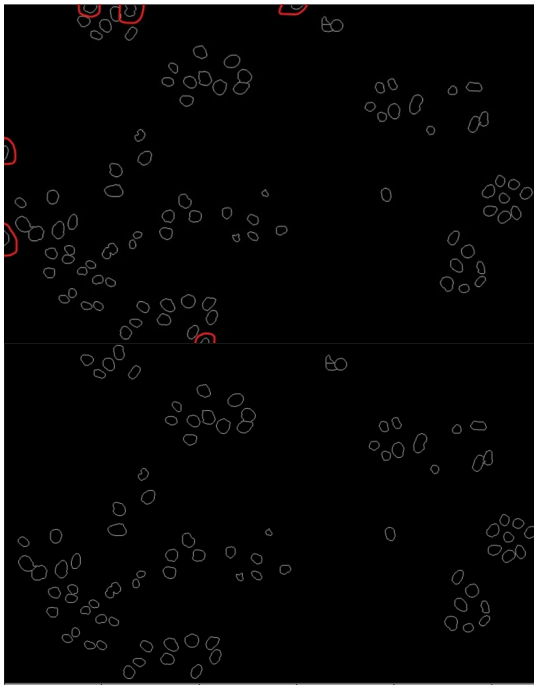

fig 3.4.1 - the top contour image is used for cell counting as it includes the cells on the border (outlined in red circle), and the bottom contour image intentionally ignores those cells so that to calculate the cell average pixel area

3.4.2 Trace and Travel Distance

For the first image, the trace and the travel distance will be initialised with zero for every cell.

Starting from the second image, when the cells are matched, the cell in the later image will inherit the trace of the cell in the earlier image. The displacement from the earlier image to the later image will also be added to the trace. The travel distance will also be inherited and updated with the displacement between the two images. The trace of the cells will be printed the same colour as the cell to show the travel history of the cells. The average travel distance will also be calculated and printed on the output image (fig 3.4.1)

3.4.3 Split Detection

The split detection is also based on the shape and the tracking result. By observation, the general case of cell splitting sequence is started by a sudden change in the shape of the cell (mainly reducing its area to a bar-like cell),

then this cell is divided into two new cells. To quantify such sequence detection, the program will analyse three properties (area, shape change, and cell existence) of each cell after the tracking result of each new frame is obtained and compare one specific cell's information in the current frame with previous frames. After this, a continuous record of such a cell and its surrounding situation within a specific time will be used to evaluate cell splitting detection.

3.4.3.1 Area Change

The first and most obvious property of a splitting cell is the sudden area change before the cell is actually splitting. This general case of sequence can be shown in the figure below (fig 3.4.3.1).

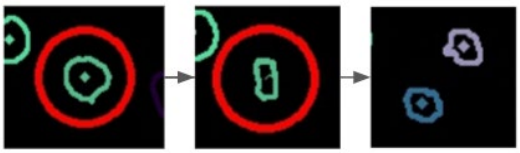

fig 3.4.3.1 - a cell splitting sequence with dramatic area change

In this case, the program will monitor each tracked cell's pixel area value and treat a cell as the potential splitting one if its pixel area value experienced a dramatic change.

3.4.3.2 Shape Change

The second property analysed for a potential splitting cell is the shape change. While there will be area change for all general splitting cells, it is possible that such change might not be as dramatic as others, but its shape change is relatively more noticeable, as shown below (fig 3.4.3.2).

fig 3.4.3.2 - a cell splitting sequence with shape change

The program will compare shape changes of a cell along a timeline by quantifying the similarity of the same cell over time using Fourier series representations of the shape [9]. If there exists a cell that experienced such sudden shape similarity change in previous frames, then such a cell is also treated as a potential splitting cell (if not already according to the area change).

3.4.3.3 Cell Existence Change

Finally, by summarising the number of new cells and the disappearance of the targeted potential cell within a specific area around the potential splitting cell, the program will decide if this cell is/was a splitting cell along with all these conditions and mark this cell's whole splitting progress in previous frames in a red circle.

3.4.3.4 Cell Splitting Marking

Finally, after detecting a split activity, the cells that are splitting will be circled using red. The example output image (fig 3.4.1) also has an example of the splitting cell.

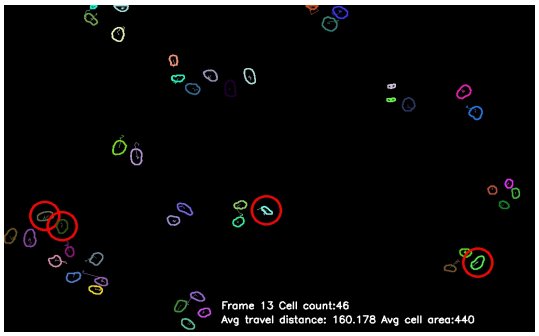

fig 3.4.1 example output with cell movement analysis information

4 Experimental Results

4.1 Cell Preprocessing and Segmentation

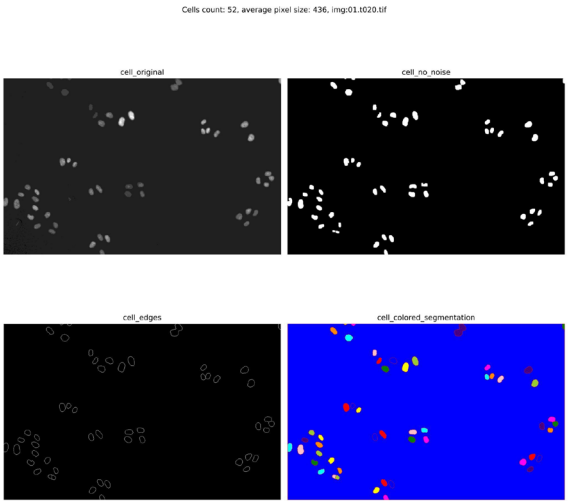

fig 4.1.1 - segmentation file comparison

For the result of cell preprocessing and segmentation, the original image is the only reference used to manually evaluate each performance of the program function purely based on human eyes, as human eyes are quite precise for the evaluation of this task specifically. More specifically, each evaluation process will sample some images among the source input images, and each sample's in-between-progress images are displayed alongside the original image for a better viewing experience, as shown in the figure above (fig 4.1.1).

At the end of each image sample and result generation, a more regional analysis is carried out to check each critical region with cells attached with each other. The final function was determined by an experimental result that handles most of the closely attached cells' segmentation while not compromising the generally isolated cell segmentation. For this program, this segmentation and preprocessing methods provide satisfying results by successfully extracting most of the cells, including those in an extremely light colour.

As for the ground truth images under the SEG folder, although few of the ground truth images provide a good reference of how the actual cell distribution should look like, most of the ground truth images under the SEG folder do not provide convincing performance. For example, among the first 10 sample SEG, only man_seg013 provides a convincing showcase, while a few of them not only cannot provide convincing results but also shows incorrect cell segmentation (as shown in fig 4.1.2).

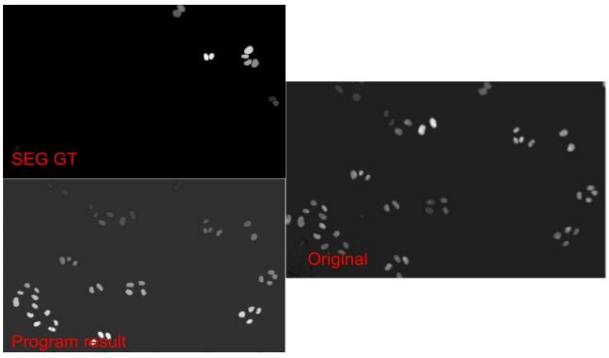

fig 4.1.2 - the result comparison for provided GT image, original image, and program processed segmentation result

Thus, based on the inconsistent performance of the provided ground truth image files, the ground truth images are not used to evaluate any cell segmentation result.

4.2 Cell Tracking

The cell tracking module is evaluated through the visual result of the random successive frames. In this part, the evaluation criteria basically examine whether the trajectories are correct for each cell despite cells in mitosis. In the discussion part, we picked up the result of frames from 26 to 35 of sequence 1 to better illustrate the tracking result and justify the correctness of tracking.

4.3 Cell Splitting Detection

The evaluation of the cell splitting detection module is based on the manual test cases created among the resulting segmented cell image. In this case, for the overall test cases generalisation, some representative cell splitting examples are manually selected from the segmentation results. An example of such test case generation is displayed below (fig 4.3.1). Then, the module function is tested among all of these cases, and the final desired function holds the parameters which cover all of the general cases and some of the non-extreme exceptional test cases.

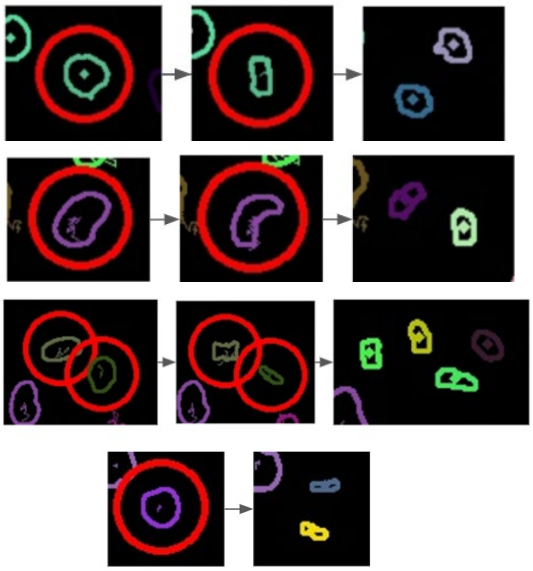

fig 4.3.1 - four representative cases collected for cell splitting sequences

5 Discussion

5.1 Cell Preprocessing and Segmentation

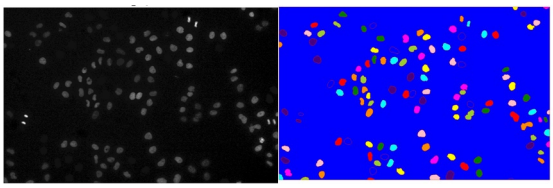

fig 5.1.1 - another demo of current segmentation result

By visual examination, the overall performance of the preprocessing can extract nearly all of the cells with some exceptions, like some closely attached cells (as shown in fig 5.1.1). However, after the final segmentation module is determined, it is still possible that some other mechanisms can be applied to more clearly separate the closely attached cells then segment them more accurately. For example, if the general shapes of two attached big cells are quite clearly outlined in the original input image, the gradient factor field of such two cells could be generated to help correct the desired centroid of such two attached cells. With the gradient vector field generated corrected centroid, combined with the current module, a more optimised version of the segmentation module could be gained.

5.2 Cell Tracking

As mentioned before, the result of tracking is successful. We picked up frames from 26 to 35, and nearly all the cells without focusing on cells in mitosis are tracked successfully based on the algorithm, and the trajectory is the same colour with the cell's contour as we can see. Here we displayed part of the result images from frames 26 to 35, as fig 5.2.1 shows. In fig 5.2.1, the trajectories of three cells are shown clearly. However, still, some problems are occurring in the tracking. As fig 5.2.2 shows, some cases of overlapping/under segmentation caused the problems, and the algorithm cannot distinguish them properly, as we only record the information of missing cells for a few sequence frames. Actually, for some cases, the overlapping/under segmentation state can last for a long time, or the change is too significant to detect. While the algorithm can distinguish some cases successfully, as figure 5.2.3 shows, which does not last for a long time. The overall tracking is at high accuracy on the tracking task.

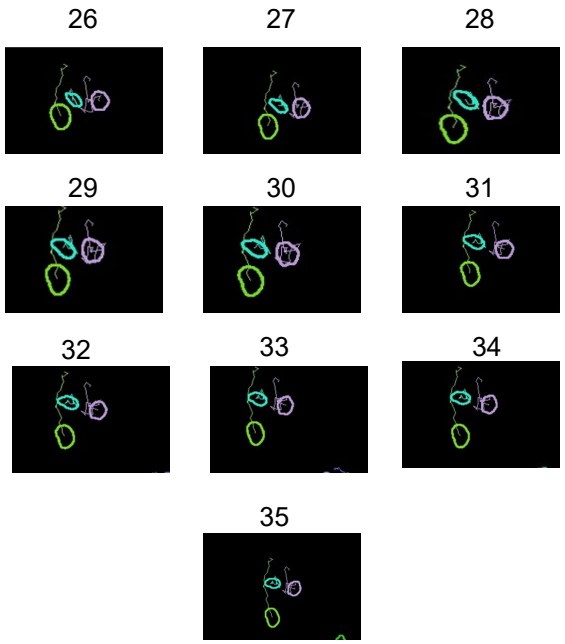

fig 5.2.1 - partial result of tracking

fig 5.2.2 - under segmentation

fig 5.2.3 - successful cases of tracking missing cells

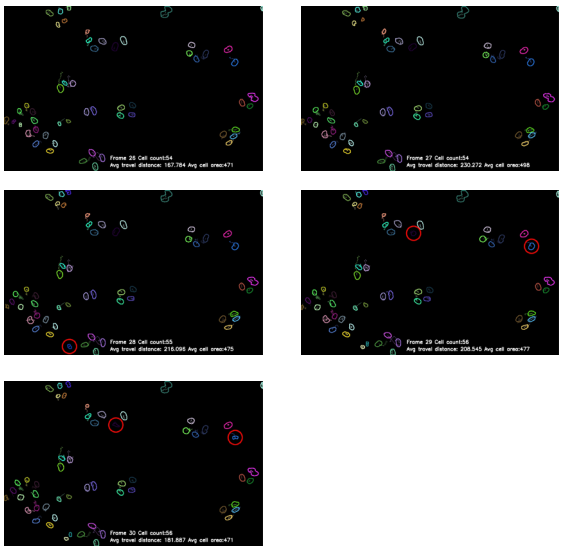

fig 5.2.1 - tracking from frame 26-30

5.3 Cell Preprocessing and Segmentation

As what is mentioned in Part 4.3 and fig 4.3.1, the cell splitting detection module can successfully detect and point out the general cases in the red circle. This is based on the assumption that three properties like area, shape, and cell amount within a certain radius must experience a quite dramatic change, and such change should occur right before splitting action. However, it is possible that all of the three properties will go through a special sequence and confuse the cell splitting detection module.

fig 5.3.1 - the failed detection of cell splitting which is circle in red

In this case (fig 5.3.1), although the actual cell splitting action is relatively clear in human eyes, for the program module, this is quite unclear. The main reason behind this is that all of the three properties sequence analysis for cell splitting all failed. More specifically, from the second frame to the third frame, the cell did not experience any dramatic change in either overall pixel area value or overall shape. The overall displacement of such a cell over this time period was also minimal, and could not make the tracking module successfully recognize the new cell and the old cell as two different cells. Instead, the tracking module treated them as the same cell moving from one place to another. Such a failure result is because the cell splitting detection module follows a quite strict checking sequence according to the behavior of cell dividing action.

In summary, the performance of the current cell splitting detection module is satisfying while handling the general cases where the splitting actions are more common.

6 Conclusion

Through this project, a series of computer vision methods were applied to process the cell images. The normalisation, contrast stretching and adaptive threshold were used to remove noise and prepare the image for the segmentation process. Watershed was used for segmentation and then used to find the contour of cells. Preprocessing generated excellent input images for segmentation. Segmentation was successful as it separated most cells with minor exceptions.

Tracking was implemented using the Hungarian algorithm to match the cells between adjacent frames. Cells matched had their

trajectories recorded. Cell count, cell area, tracing and travel distance Cell splitting detection were implemented too. In general, tracking and splitting detection were working as specified by the project specification.

Some future scope including:

- Improve segmentation with gradient vector fields to separate cells which are close to each other.
- Record and use the centroid of cells before applying the watershed algorithm.
- Use machine learning-based methods to conduct cell segmentation

7 Group Contribution

Qianyi Li: Li is in charge of the tracking module in the work based on the segmentation result. Li provided the framework of the project and helped with the part of segmentation. Li also gives valuable suggestions for splitting detection.

Ziqiao Lin: Lin is in charge of image preprocess optimization. Lin implemented the cell segmentation with its corresponding evaluation proposal. Lin also helped with the method proposal and implementation of cell splitting detection based on Li's work on cell tracking.

Sihang Wang: Wang's contribution in the experiment part is the preprocessing and segmentation part. As for the report, Wang is in charge of the report structure and started the draft report, including the introduction and methods part.

Yuchao Fang: Trials with preprocessing and segmentation, construction part of tracking and some helper functions, testing segmentation and tracking. Works on conclusion and demonstration in report and presentation.

Fei Yu: Contributes to the experiments in cell splitting and result analysis. For the report, Fei helped complete the methods part. Also in charge of the making of demo videos and related slides.

References

[1] E. Meijering, O. Dzyubachyk, I. Smal, W. A. van Cappellen. Tracking in cell and developmental biology.Seminars in Cell and Developmental Biology, vol. 20, no. 8, pp. 894-902, October 2009.

[2] Min Liu, Yue He, Yangliu Wei, Peng Xiang, Plant cell tracking using Kalman filter based local graph matching, Image and Vision Computing, Volume 60,2017, Pages 154-161.

[3] https://www.tutorialspoint.com/opencv/opencv_adaptive_threshold.htm

[4] https://www.geeksforgeeks.org/python-opencv-cv2-erode-method/

[5] https://www.geeksforgeeks.org/erosion-dilation-images-using-opencv-python/

[6] https://www.sciencedirect.com/science/article/pii/S1084952109001517?via%3Dihub

[7] Dewan M M, Ahmad M M, Swamy M M. Tracking Biological Cells in Time-Lapse Microscopy: An Adaptive Technique Combining Motion and Topological Features[J]. IEEE transactions on bio-medical engineering, 2011, 58(6):1637-47.

[8] J. Yan, X. Zhou, Q. Yang, N. Liu, Q. Cheng and S. T. c. Wong, "An Effective System for Optical Microscopy Cell Image Segmentation, Tracking and Cell Phase Identification," 2006 International Conference on Image Processing, 2006, pp. 1917-1920, DOI: 10.1109/ICIP.2006.313143.

[9] https://pyefd.readthedocs.io/en/latest