Tracking Cells and Detecting Mitosis Events in Time-Lapse Microscopic Image Sequences

Victor Chan, Zixuan (Leila) Yuan, Yu Jie Lin, Peter Liu

I. Introduction

The tracking of cells plays an important role in medical biology. In particular, the study of movements, division, and interactions between cells is a field that is still yielding active research each year. Recently, due to the amount of available live cell imaging data and the advance in image recognition methods, researchers are starting to use more sophisticated methods, including machine learning and computer vision techniques, to track and analyse cell movements effectively. To achieve this, there are mainly two tasks to be accomplished.

The first task involves segmenting cells in an image sequence. In typical scenarios, cells to be studied are often infused with fluorescent protein and placed on a petri dish. Digital images are then taken to be processed on a computer. To do this, various segmentation methods are used to extract cell contours out of the image.

The second task involves tracking cell movement. After different cells are obtained from an image, effort is required to match cells across different frames, forming their trajectory in the temporal domain. Once these are established, various analysis can be performed.

In this task, we are given four sequences of cell data to study. The first question requires segmenting and tracking cells through time, and the second question involves calculating certain values for the frame. This will involve finding the cell count in the image, averaging the size of cells segmented, and calculate the average displacement of cells. We are also asked to find the number of cells currently dividing(mitosis).

Each dataset contains 92 frames of single-channel, 16-bit 700 x 1100 images. Each of these sequence are taken in different lighting conditions with a varying number of cells. In all these images, the cells themselves are of a similar colour and intensity as the background. For this reason, sophisticated pre-processing segmentation techniques will be warranted to properly extract individual cells from the background.

Before proceeding to the task, we reviewed various literature to understand the state of biological imaging computer vision techniques.

II. Literature review

A. Image segmentation and contouring

Image segmentation refers to a well-researched computer vision technique which separates wanted objects from background and partition them[1] with respect to features and properties. It is an essential branch of digital image processing as it helps to reduce the complexity of an image before any future analysis is performed. It is widely used in many notable areas including image-based search, medical imaging, face recognition, number place identification etc. In this case, image segmentation is imperative to separate cells from the noisy background to simplify further operations. After reviewing relevant literature, we have shortlisted a few classic algorithms for image segmentation, and briefly discussed their merits and shortfalls.

1) Thresholding: Thresholding is considered one of the most basic approaches to segment an image. In this process, a threshold pixel intensity value is selected. This value is used to compare with each pixel in the image, dividing them into parts lower and higher than the threshold value. This binary classification gives us the foreground and background respectively.

In order to select an optimal threshold t, one way is to use Otsu's thresholding algorithm. Otsu's algorithm follows the idea of exhaustively searching for a threshold that maximises the inter-class variance. In other words, given a candidate threshold t*, pixels are separated into two parts. When the between-group variance reaches a maximum, as in

$$\partial_B^2 = p_0 p_1 (\mu_0 - \mu_1)^2$$

t is selected as t*, where p_0 represents the fraction of pixels below the threshold, p_1 represents the fraction of pixels above the threshold, μ_0, μ_1 represents the mean of two separated classes. [1] [2]

Due to its simplicity, thresholding is a commonly used strategy. However, there are a few notable limitations. For example, it relies heavily on images having clear separable boundaries and a bi-modal nature of the histogram. [3] For images not taken in well-controlled scenarios, this method is often prone to error. In this particular task, using simple thresholding can be problematic due to the presence of noise in the image, as well as some cells having overlapping boundaries, adding to the complication of the task. Uneven lighting and fluorescence to the image often lead to some cells falling under the threshold, making this method alone unreliable to produce good segmentation results.

2) Predetermined cell intensity profiles: Another method proposed is to use template matching where we fit predetermined cell intensity profiles as a mask to the cell image. Using the profiles as templates generally works well when cells have consistent shapes. It is much less accurate with cells with more significant shape variations or when cells are merged into peculiar shapes, which are the cases we need to account for in this task.

3) Watershed transform: Watershed transformation is also an image segmentation method that operates on a grayscale image and divides it into different regions topographically. It treats the pixel intensity as altitude, where local intensity minima segment the image into contours.

However, this method is subject to noise, and it may result in over-segmentation which can mean a single cell may be cut into multiple segments which is not ideal in our project.

4) Deep learning: Cell segmentation in biological applications often uses deep learning to achieve satisfactory outcomes. DeepCell and U-Net are two most notable models which are based on convolutional neural networks. [3] DeepCell is a CNN framework designed to fit the purpose of segmenting biological images by using a patch-based, single pixel classification objective. [3] U-Net resembles encoder–decoder-style neural network that reconstructs the image through manipulating spatial resolution and handling the representation encoding with the help of multiple convolutional layers.[3][4]

These two modals exhibit high accuracy in cell segmentation, with fewer errors of failing to distinguish merged cells compared to other networks including CellProfiler, which uses Otsu thresholding, and unclumping as the core technique. [3] In general, both of them are effective at recognizing boundaries hence has the ability to perform contouring. [3]The diagram below shows the rate of having incorrect segmentations for split and merged cells, followed by two illustrations of errors.

Despite the fact that deep learning produces great accuracy in cell segmentation, the computational cost of training a model can be expensive, with an additional requirement of having a large dataset.

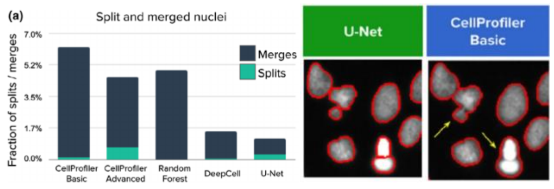


Fig.1. Results collected to evaluate the performance of deep learning models. [3]

B. Tracking and cell association

The process of cell tracking involves two main steps: Segmentation and association. The time-lapse images need to be segmented in a way that the individual cells can be separated from the background. Cell association is then needed to connect segmented cells between images to ensure the same cell is being tracked from frame to frame.

One of the simpler ways for cell association is to find the spatially nearest cell between frames, using the centroid position of cells to determine distances [5]. However, this can result in errors and mismatches when it comes to images that have many cells, where a different cell may move to become closer to the centroid of the target cell, or for rapidly moving cells which may cause cells to move too far from the cell in the previous frame to be considered nearest.

A solution to this is to consider other factors on top of spatial distance, such as average intensity, area, perimeter, orientation, boundary curvature, estimated displacement, etc. The more features that are used for comparison, the less ambiguity there would be during cell association.

Some cell segmentation methods can also be used in cell association. Template matching can help with image registration between frames, registration referring to image alignments using either cell intensity or geometric features [5]. Deformable models can also be used to perform live cell association by using segmentation results in one frame as the initialisation for segmentation in the next frame [5]. This would work well for images with lower cell density (less cells in the frame) and the speed of cell movement is not too high (at most travelling their diameter's distance between frames).

More complex methods may be needed for cell association such as gradient-vector flows, estimated cell dynamics, or probabilistic tracking methods.

C. Mitosis detection

Mitosis refers to the process in which a mother cell splits into two or more cells which are referred to as daughter cells. To detect the process of mitosis and the respective daughter cells produced requires some additional methods. The process of mitosis detection itself can be split into three parts: identification, localisation, and segmentation.

Identification will involve detecting whether mitosis is actually occurring in the image, localisation methods will help localise the mitosis division process spatially and temporally, and segmentation of the mitosis process into one of its four stages: Interphase (before split, still looks normal), start of mitosis (cells shrink), formation of daughter cells (two or more cells visibly attached to one another), and separation of daughter cells. Due to frame rate constraints and speed of the process, not all the phases will be captured by the camera.

In terms of methods used in mitosis detection, there are several methods, namely tracking base methods, tracking free methods, and hybrid methods.

1) Tracking: In this method, one method uses cell trajectories from cell tracking, combined with other properties such as area, perimeter or circularity gained from adjacent frames, to determine whether mitosis occurred.
Another method uses specialised mean-shift kernels to determine which cells are undergoing mitosis during the tracking stage [6]. This method uses established cell trajectories to work backwards in the time dimension, detecting a mitosis event when the centroids of two daughter cells become close enough to be merged.

A segmentation-driven approach has also been used, using a framework consisting of a forward-tracking and a backward-tracking process [6]. The forward-tracking process is used to generate the standard cell tracking results, while the backward-tracking generates a sort of tree structure, with each branch representing a mitosis event.

A final method proposed was a Kalmen filter which predicts the movement of cells and searches for mitosis events in a neighbourhood around the predicted areas [6]. This method uses both the segmented cell boundary and the area size of cells at the predicted positions to determine if mitosis has occurred.

The main issue that arises with tracking based methods are derived from issues that come from cell tracking.

2) Tracking free methods: The use of Support Vector Machines (SVM) is one of the methods for mitosis detection that doesn't use tracking [6][7]. This method involves pre-processing of the image, extracting the target spatiotemporal (of both space and time) volume using the volumetric region growing method, and using SVM to determine whether mitosis occurs within each frame of the target sequence.

A convergence index filter can be used to find nuclei regions in the input, and the shape characteristics of the nuclei regions are used to train a linear classifier for mitosis detection.

Another approach uses a nonnegative mix-norm convex optimisation problem for the sparse representation of mitotic cells which is solved using a tailored online learning algorithm [6]. An SVM then uses the sparse representation for mitosis detection.

As tracking free methods do not rely on the movement tracking, they are more suited for static images. However, as our input data is time-lapse data, ignoring the temporal aspect of mitosis detection, which can aid in creating a more robust mitosis detection method.

3) Hybrid methods: Hybrid methods combine both the visual aspect of mitosis events, as well as the temporal information from time lapse sequences, to overcome the downsides of tracking and tracking free methods. In hybrid methods, graphical models are used to discover conditional dependence structures between random variables. This allows us to identify mitotic sequences, but it also can allow us to know the exact time daughter cells are created.

There are three main steps in hybrid methods: Candidate sequence extraction, feature extraction, and mitosis detection with the graphical models.

III. Methods

A. *Preprocessing and segmentation*

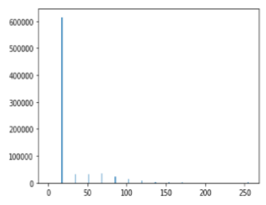
Due to the similarity of the cell intensity with the background, pre-processing is needed to ensure the cells can be more easily detected. We tried out a few methods, such as using min-filtering followed by max-filtering, then a subtraction to remove the background, as well as contrast stretching the input image and applying otsu thresholding.

However, these methods had some issues that clashed with our cell detection and cell tracking. Min-max filtering method to remove the background was useful highlighting more of the cells, however, in removing the background, parts of cells were also removed which resulted in our detection algorithm detecting multiple cells where there was only one. Otsu thresholding was also not the best as the threshold it chose often leave out less intensive cells, which is quite common in a few datasets.

For this reason, we decided to define a custom thresholding algorithm for this task to determine the best threshold value to pick. This custom algorithm will find the intensity value of the background (which in the case for all the input sequences are lower than the intensity of the cells) to find the optimum threshold value to separate the cells from the background.

After applying contrast stretch on the input image, we get the resulting histogram of intensities in fig 2.

We know the tall line represents the background intensity as the most common intensity value will be the background. With this information, the threshold is set to be slightly above this value, so only pixels lower than this value will be removed. Since this method often includes some background noise, a gaussian blur is applied, followed by morphologically closing the holes and applying erosion to shrink some mildly touching boundaries. This operation enables most small flecks and noise to be blended into the background.



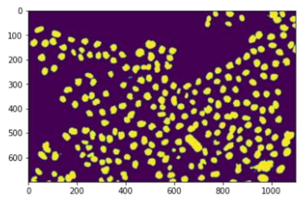


Fig. 2. Colour intensities of contrast stretched input

Fig. 3. Output of preprocessing

In the detection stage, we use the inbuilt findContour function from openCV to extract the contours from our pre-processed image. When extracting contours, however, we only aim to look at those greater than a predetermined size set at 10% of an average cell. From our observation, if a cell's extracted contour is smaller than this threshold, it is normally of very weak intensity. In most cases, the contour is very uneven. Therefore, we decided that cells below this boundary shall not be included in our calculations.

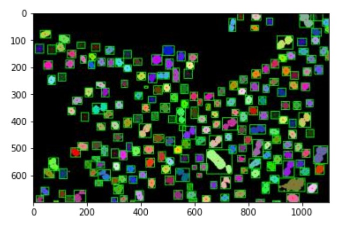


Fig. 4. Detection of cells

B. *Cell Tracking*

The tracking of cells across frames can be viewed as a combinatorial optimization problem. For every cell in a subsequent frame, we need to match it with one of the cells in the previous one, or discard it if such matching is unlikely. Adding in the possibility of mitosis, the problem resembles one of travelling postman problems. These problems are NP-complete, where obtaining a deterministic solution is often not possible due to the exhaustive search space when more variables come into play.[8] Non-deterministic solutions involve iteratively trying random paths and keeping record of their results, but they could lead to different answers in subsequent trials. For this reason, we have decided to use our own method.

Since the use of brute-forcing combinations would result in incredibly long processing times, we have to simplify the question by making some assumptions. We borrowed an idea of Occam's razor which says that the simplest explanation that involves fewest complications is usually the best one. Following this, a measure of similarity of cells is defined. In our similarity metric, we measure three separate values related to centroids position differences, area discrepancies, and the change in speed.

Distance between cells is defined as the Euclidean distance between x and y coordinates of the centroid:

$$Dist(c_1, c_2) = \sqrt{(p_1 - p_2)^2 + (q_1 - q_2)^2}$$

Area of a cell is simply defined as

$$Area_{c1} = area\ in\ contour\ c_1$$

Change in speed of a cell is defined as the exponential moving average of the previous change in speed, added to the latest distance change. This value is initialized as 1.

$$\Delta Spd_{c1f0,\ c1f1} = \Delta Spd_{c1f0}\,(1 - v) + Dist\big(c_{1f0}, c_{1f1}\big).v$$

Finally, the similarity value of any two cells c_0 and $c_1$is defined as:

$$S_{c1,c2} = \alpha.Dist(C_1, C_2) + \beta|\,Area_{c1} - Area_{c2}| + \gamma\Delta(Spd_{c1,c2})$$

where α, β, γ are sensitivity parameters to be optimized.

This formula is useful in measuring the similarity of two cells because it not only puts the distance and area discrepancies between two cells into consideration, but also accounts for a sudden change in speed if the two cell is identified as a pair. In other words, the lower the similarity value is, the more similar the two cells look like. The aim is to find a matching cell that is spatially close to one in the previous frame, with a similar area, and with a similar velocity. This comes with the assumptions that most cells do not travel far between frames, the area does not change much (except during mitosis), and the velocity stays relatively constant.

From this formula, we obtain a similarity matrix between all cell pairs in the current frame and the previous frame, where there are x cells in frame 2 and y cells in frame 1 respectively:

$$SimMatrix(f1, f2) = \begin{bmatrix} S_{11} & S_{12} & \cdots & S_{1x} \\ S_{21} & S_{22} & \cdots & \\ \vdots & & \ddots & \\ S_{y1} & & & S_{yx} \end{bmatrix}$$

We can then find the index of the minimum value for each column to obtain an array BestMatch(f1, f2), matching the most similar cells between the current frame and the previous frame:

$$BM(f1, f2) = Argmin(SimMatrix(f1, f2), axis\ 0) = [m_1 \quad m_2 \quad m_3 \quad \dots \quad m_x]$$

There will be cases where values in the Best Match matrix are not unique, indicating multiple cells in the current frame are very similar with the same cell in the previous frame. In this scenario, there are mainly two possibilities: The two (or more) cells matching with a single cell are the children of that cell (i.e. mitosis occurred), or one of the cells is the true match and the others shall be matched with a different cell. The first scenario is a factor used in mitosis detection (which will be detailed in the mitosis detection section). If the mitosis possibility is ruled out, we will delete the row and column of the successful match, then re-calculate the Best Match matrix, until every value in the Best Match matrix is unique.

Once we match the cells between the frames, we draw the cell in the current frame to have the same color as the cell in the previous frame, and we also draw the path that cell has travelled.

C. Mitosis Detection

When mitosis occurs, we observe the three stages a cell undergoes during mitosis: an interphase, a phase marked by acute shape change, and a final phase splitting into daughter cells.

From this we have made a few observations:

- the cell undergoes a large decrease in area during the start of mitosis
- the daughter cells produced are roughly the same size
- the midpoint between the daughters is relatively close to the centroid of the father cell
- and the daughter cells distance to the father is relatively uniform.

From these observations, we come up with another equation for mitosis, measuring the likelihood of cell 1 splitting to cell 2 and cell 3 in another frame:

$$M_{c1,c23} = \alpha.Dist(C_1, Midpt(C_2, C_3)) + \beta|\,Area_{c2} - Area_{c3}| + \gamma.\Delta(Area_{c1})$$

where α, β, γ are yet another sensitivity parameters to be optimized. In this equation, the mid points of cells are defined as

$$MidPt(c_1, c_2, \dots c_n) = (\frac{\sum_{i=1}^{n} p_i}{n}, \frac{\sum_{i=1}^{n} q_i}{n})$$

With the mitosis cost function, the lower the value, the more likely the cells (C1, C2, …) are to be daughters. This cost function penalises based on the distance from the parent to the midpoint of the two cells, difference in area between daughters, and recorded change in area of the parent from the previous frame.

Back in the similarity matrix, whenever there is a duplicate in the Best Match matrix, we have to consider how likely a mitosis has occurred. A matrix for this mitosis scenario, $MS_{xy,123,1}$ is calculated:

$$MS_{xy,123,1} = \begin{bmatrix} S_{1x} & M_{x12} & M_{x13} \\ S_{1y} & M_{y12} & M_{y13} \end{bmatrix}$$

This matrix calculates all the possible outcomes of cell 1 in this scenario, either matching with cell x, cell y, or be a product of mitosis with another cell (2 or 3) from father cells x and y. In order for mitosis to be considered valid, the mitosis value will be compared with the similarity value. If a heavy area change has occurred for the father cell, the mitosis cost could go to negative, which will inherently put priority to all mitosis possibilities with that cell, since similarity value is always non-negative.

Computing this matrix for all cells in conflict in frame 2 give us:

$$MS_{xy,123,123} = \begin{bmatrix} S_{1x} & M_{x12} & M_{x13} \\ S_{1y} & M_{y12} & M_{y13} \end{bmatrix} \begin{bmatrix} S_{2x} & M_{x23} \\ S_{2y} & M_{y23} \end{bmatrix} \begin{bmatrix} S_{3x} \\ S_{3y} \end{bmatrix}$$

Or simply

$$MS_{xy,123} = \begin{bmatrix} S_{1x} & M_{x12} & M_{x13} & S_{2x} & M_{x23} & S_{3x} \\ S_{1y} & M_{y12} & M_{y13} & S_{2y} & M_{y23} & S_{3y} \end{bmatrix}$$

It will be logical to assume that one cell in frame 1 can only lead to one outcome in frame 2. Therefore, we can only pick one value in one row.

At this stage, various programming techniques can be used to find the minimum overall cost. For example, dynamic programming is a well-tested method for finding the answer of such 2-dimensional optimization problem. After some experimentation, we observe that choosing minimum values and eliminating pairs is extremely quick and nearly always yield the same result as dynamic programming. Therefore, we have instead used this simpler and more efficient approach to matching outcomes in this scenario:

```
For i in num(rows) do:
    1) Match the next pair with the lowest value in MS
    2) Cross out all the rows and columns associated
```

In reality, this is a more direct method and allows the algorithm to be expanded to include more possibilities without scalability issues.

This mitosis settlement is only triggered when a conflict is found in the similarity matrix. Once the conflict is settled, the matched pairs are excluded from $SimMatrix$, and BM is recalculated to reflect changes. The 1-to-1 matching will then take place again.

D. Analysis of Cell movement

In a separate task, cell movements are analyzed and displayed in a separate window. To do this, we keep a list of all the cells in each frame. The average displacement of all cells can be derived. In the cell class, we keep a variable which shows the current position of the cell, as well as a history list recording all of the old positions it appeared in previous frames. For all cells in the current frame, we calculate the moving distance by applying the same Euclidean distance formula of its current position and latest position in the previous frame.

The average for all cells in a frame is defined as

$$AverageDisplacement(f1) = \frac{\sum_{i=1}^{n} Dist(C_i f1,\ C_i f0)}{n}$$

And the number of dividing cells is obtained as

$$DividingCells(f1) = \#Cells\ matched\ in\ mitosis$$

Since the history list tracks all positions of a cell from the beginning to the current frame, this allows us to draw a trail of its moving trajectory, so that visual inspection can be conducted. For clarity reasons, only the last 15 positions were drawn in the outputs.

IV. Experimental Setup

A. Setup

Computations and rendering are done in batches for all sequences on a desktop computer. Since the approach we used is not very computational heavy, a GPU is not needed for the rendering.

TABLE I. SETUP

	Component		
	CPU	***RAM***	***Language***
Model	i7-8700	32GB DDR4-3200	Python 3.7.4 OpenCV 3.4.2

V. EVALUATION

A. Cell Detection

One of the most widely used methods to evaluate the performance of cell detection is Intersection over Union (IoU):

$$IoU = \frac{Area\ of\ overlap}{Area\ of\ union} = \frac{|True \cap Pred|}{|True \cup Pred|}$$

After visually inspecting some of the ground truth data provided, we realized most of the data are not fit to use this method. Instead, we apply contrast stretching and gamma correction to each image, and inspect the result of our segmentation to dozens of selected frames, distributed evenly across all of the four sequences.

B. Cell Tracking

We extract all the cell labels to the ground truth datasets, and developed a program to check if the cell centroids and contours in our output are located on these labels. We also visually inspect the moments of the cell trails to analyze them qualitatively.

C. Mitosis Detection

Similar to cell segmentation, we treat the evaluation of mitosis detection as a multi-class classification problem. In this scenario, experimental results are classified into True Positives(TP), False Positives(FP), True Negatives(TN) and False Negatives(FN).

$$Accuracy = \frac{\#TP + \#TN}{\#TP + \#FP + \#TN + \#FN}$$

$$Recall = \frac{\#TP}{\#TP + \#FN}$$

$$Precision = \frac{\#TP}{\#TP + \#FN}$$

When testing, we visually inspect all the mitosis detected by the program, and judge whether it is a correct classification by navigating earlier and later in that sequence.

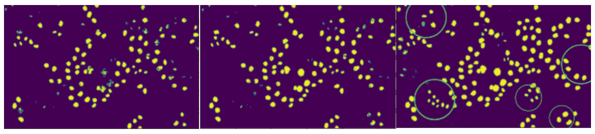
VI. RESULTS AND DISCUSSION

A. Cell Detection

TABLE II. CELL DETECTION

Sequence	Metric			
	Frames	***Ground Truth***	***Missed***	***Accuracy***
Seq01	91	8731	1482	83.0%
Seq02	91	5355	713	86.6%
Seq03	91	2356	266	88.7%
Seq04	91	3417	314	90.8%

As shown in the table above, we obtained satisfactory results for cell segmentation with an accuracy of 87.28% in all four data sequences. The result was achieved by using image stretch, customized thresholding and followed by a watershed transform. One problem we faced during implementation was that due to the nature of thresholding, cells that are under the threshold intensity are ignored. It was improved by introducing a new threshold for each data sequence, as shown in the three images below.



(a) Contrast stretch + Otsu (b) Min-max filter + Otsu (c) Contrast stretch + custom threshold
Fig. 5. Cell segmentation with different methods

However this issue is not completely solved, occasionally we still have cells that are missing from segmentation as seen in Fig. 5(a). Another issue is that the boundaries for overlapping cells could not be distinguished completely (seen in Fig. 5(b)), hence resulting in treating multiple merged cells as one cell only. This issue was improved by refining segmentation, however cannot be completely avoided.

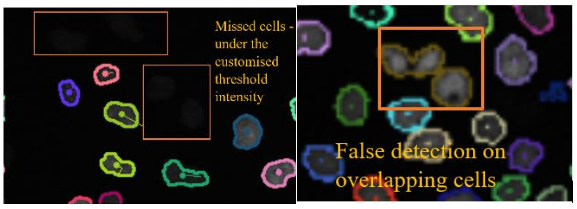

Missed cells - under the customised threshold intensity
False detection on overlapping cells

Fig. 6. Cell detection errors

Machine learning-based methods can help to improve this issue to an extent. Based on the research we conducted, the boundaries of objects are more precisely detected and mapped with the help of CNN models such as U-net, deepCell etc. Another way to improve is to have a predetermined cell intensity profile to mask over the image, so that cells with standard shapes can be detected more precisely. However it might not be the ideal solution due to the variating nature of cell shapes.

B. Cell Tracking

TABLE III. CELL TRACKING

Sequence	Metric		
	Total	***Centroid***	***Contour***
Seq01	8731	69.9%	83.0%
Seq02	25512	43.9%	86.3%

We observe that the measure of contour is a much more forgiving metric than centroid. While most of the labels are detected in the contours, a much smaller percentage are tracked precisely at the centroid. We attribute this to different pre-processing techniques, including filtering, thresholding, and morphological transformations. When some of these labels are not found at the centroid, they are often quite close to it, usually just a few pixels away. Other labels that are not found at the centroid and contours happen to be cells lost in segmentation and thresholds.

The below screenshots demonstrate the base case of tracking across two frames. For each individual cell, the centroid is highlighted, with a thin trail connecting to the centroid of it in the previous frame. A typical example can be seen in the blue cell highlighted below, where the trailing line shows the trajectory of the cell.

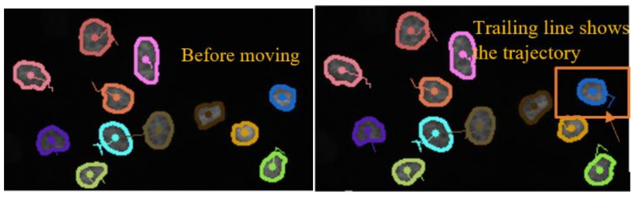


Fig. 7. (a) base case for cell tracking (b) Trails

The accuracy of cell tracking is usually affected by false cell segmentation and detection. As shown in the below Figure. 8(a) and (b), we see that the purple cell is lost in the next frame due to the reduced intensity. Overlapping cells also introduce problems. As shown in the Figure. 8(c) and (d), two separate cells in the previous frame can be wrongly merged and tracked as one cell once they become too close.

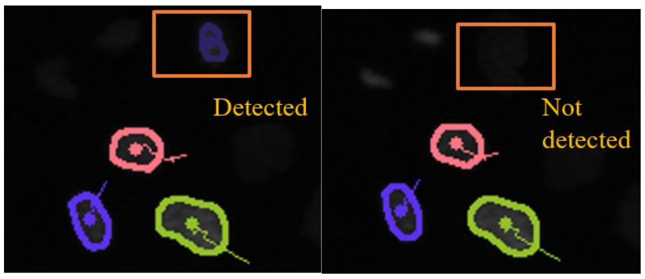


Fig. 8. (a) (b) Segmentation errors

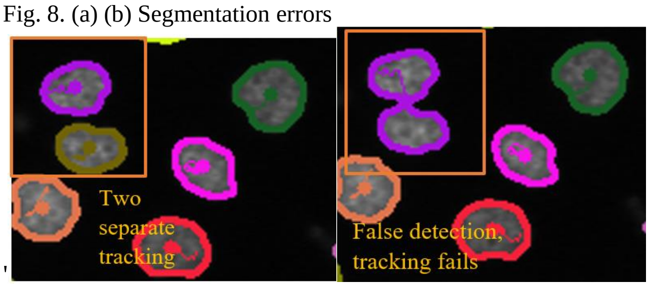


Fig. 8. (c) (d) Overlapping cells

We conclude that cell tracking can be significantly enhanced by improving cell segmentation and detection. Improvements for detection were already discussed above.

C. Mitosis Detection

TABLE IV. MITOSIS DETECTION

Sequence	**Metric**		
	Accuracy	***Precision***	***Recall***
Seq01	99.6%	86.7%	89.7%
Seq02	99.6%	90.6%	82.9%
Seq03	99.5%	90.2%	86%
Seq04	99.8%	82.5%	93.1%

The accuracy value for mitosis detection is close to 100% across all images. However, this metric provides little insight into how well mitosis events are detected, due to the domination of true negatives in this task. Still, the precision and recall values indicate that the method performs quite well in all these sequences, achieving more than 80% across the board.

When the rendered results were inspected more closely, it is noted again the need for good segmentation data for mitosis detection to work well. For example, the method obtains a high recall score in seq01 and seq04, when segmentation provides satisfactory cell information for mitosis calculations most of the time. When we move to seq02 where a lot of cells are present starting from frame 1, mitosis often results in overlapping cell contours. In sequence 3, the tricky lighting condition led to subpar cell information obtained across frames, which hindered the overall mitosis detection efficacy.

On the other hand, the precision score is hampered by some false positives (shown in Figure. 10(a) and (b)), which is a result when cells clump in a small region. In these places, brief touch and separation of cell contours happens frequently, which may trigger a false mitosis detection . This is especially evident in Seq04, which has the largest number of cells in the dataset. The precision score is also reduced by false negative cases when a father cell is detected only for one frame, which the program will not register any area change for that cell, decreasing the likelihood of it being classified correctly.

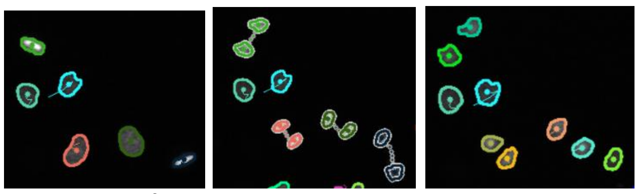


Fig. 9. Successful mitosis detection example

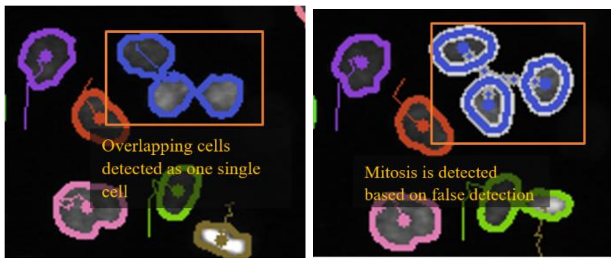

Overlapping cells detected as one single cell
Mitosis is detected based on false detection

Fig. 10. A false positive case for mitosis detection

Mitosis detection can be improved by expanding the robust searching around cells. By using convexHull, we can increase the number of candidates, which in turn expand the matrix so that more comparisons can be made, hence having a more accurate mitosis detection. On top of that, we can also include more history frames in the cell class instead of only analysing the previous one, so that a more extended evaluation can be made on the movements.

To achieve this, we can store all unmatched cells from previous frames and re-use them in the similarity matrix. To add temporal difference into the question, a penalty shall be introduced for each cell, where the value is proportional to the duration of frames this cell has gone missing. When overlapping between cells happens briefly, this method shall be able to retrieve the lost cell from a few frames ago, depending on how far the cell has travelled during this period. However, this could also introduce another parameter to the equation, adding to the list of parameters to optimize for.

VII. Conclusion

Cellular tracking by computer vision methods is a well studied problem. We reviewed recent advances and common techniques used in cell segmentation and tracking. Based on our observations, we introduced an original method for segmenting, tracking and detecting cell divisions in a few time-lapse sequences, and obtained satisfactory results for some of the image sequences. We evaluated the output and analysed results from multiple perspectives. For further improvements, we suggest combining our method with multiple approaches, e.g. deep learning methods and SVM-based techniques, for cell detection and tracking.

VIII. Contribution

Victor Chan (z5262347): Reading literature, drafting introduction, methods (segmentation, tracking & mitosis), evaluating results, writing discussion, drafting conclusion, improvements, video-editing, programming, formatting

Zixuan (Leila) Yuan (z5261559): Reading literature, methods (segmentation), evaluating results, writing discussion, improvements, programming, formatting

Yu Jie Lin (z5258280): Reading literature, methods (segmentation), evaluating results, writing discussion, improvements, programming

Peter Liu (z5253738): methods (segmentation), evaluating methods, evaluating results, programming

References

[1] Lecture notes - introduction, Image Processing Part 1

[2] J. Xue and D. M. Titterington, "t-Tests, F-Tests and Otsu's Methods for Image Thresholding," in IEEE Transactions on Image Processing, vol. 20, no. 8, pp. 2392-2396, Aug. 2011, doi: 10.1109/TIP.2011.2114358.

[3] Caicedo, J.C., Roth, J., Goodman, A., Becker, T., Karhohs, K.W., Broisin, M., Molnar, C., McQuin, C., Singh, S., Theis, F.J. and Carpenter, A.E. (2019), Evaluation of Deep Learning Strategies for Nucleus Segmentation in Fluorescence Images. Cytometry, 95: 952-965. https://doi.org/10.1002/cyto.a.23863

[4] Falk, T., Mai, D., Bensch, R. et al. U-Net: deep learning for cell counting, detection, and morphometry. Nat Methods 16, 67–70 (2019). https://doi-org.wwwproxy1.library.unsw.edu.au/10.1038/s41592-018-0261-2

[5] 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. https://doi.org/10.1016/j.semcdb.2009.07.004

[6] A.-A. Liu et al. Mitosis detection in phase contrast microscopy image sequences of stem cell populations: a critical review. IEEE Transactions on Big Data, vol. 3, no. 4, pp. 443-457, October 2017. https://doi.org/10.1109/TBDATA.2017.2721438

[7] Veta, Mitko, et al. Assessment of Algorithms for Mitosis Detection in Breast Cancer Histopathology Images. 2014, doi:10.1016/j.media.2014.11.010.

[8] Adleman, Leonard (1994), "Molecular Computation of Solutions To Combinatorial Problems" (PDF), Science, 266 (5187): 1021–4, Bibcode:1994Sci...266.1021A

[9] Yin, F., Makris, D., Velastin, S.A.: Performance evaluation of object tracking algorithms. In: IEEE InternationalWorkshop on Performance Evaluation of Tracking and Surveillance, Rio De Janeiro, Brazil, pp. 1-25. Citeseer (2007)