

Imperial College London
Department of Computing

Learning to Automatically Detect and Track Cells in Microscopic Imaging

by

Pedro Damian Kostelec

September 2014

Supervised by Ben Glocker

Submitted in part fulfilment of the requirements for the
MSc degree in Computer Science (Artificial Intelligence) of Imperial College London

Abstract

Automatic cell detection and tracking for *in vivo* microscopic image sequences enables the efficient analysis and quantification of cell behaviour in their natural environment, thus improving the study of drugs and our understanding of living beings. The technical achievements in *in vivo* image acquisition in the past few years has enabled us to study cell behaviour in a much more detailed manner without dramatically changing their natural environment. This research focused on the detection and tracking of cells in these images, which often exhibit motion artefacts, blurred frames, obscured cells and background noise. Traditional cell detection methods cannot efficiently analyse these images.

We upgraded a previous cell detection method that learns to classify non-overlapping candidate regions as cell or not-cell. The computation time of the original method has been significantly reduced to process each frame (of dimensions around 512-by-512 pixels) between 0.5 and 1.5 seconds. The method achieves high precision and recall values on the studied datasets and can be trained with dot-annotation – each dot corresponding to a cell. In some cases it is possible to use a pre-trained detector to detect cells in a new, previously unseen dataset and still achieve good detection rates.

The tracking method uses the centroids of the detected cells and attempts to link them into trajectories. The method relies on a global data association approach to reliably generate trajectories based on a global decision. This research has upgraded previous methods by approaching the problem of defining likelihoods of linking tracklets from a machine learning point of view. Because the likelihoods are obtained directly from training examples, the method is able to produce good results even on low quality images, where several frames in a sequence can become out-of-focus, and in which cells can disappear and reappear over time, etc.

The combination of these two data driven algorithms for cell detection and tracking has been shown to be a promising approach to automatic cell tracking. Extensive evaluation has shown that the methods achieve very good results on the studied datasets. We also present a set of possible further improvements to the methods that would further improve their performance.

Acknowledgement

I would like to acknowledge a number of individuals that have supported me throughout this project or in some way or another contributed to the development of the automatic cell tracker presented in this report.

First, I give my sincerest gratitude to my supervisor, Dr. Ben Glocker, for his continuous feedback, guidance and contagious motivation for the project.

Second, I would like to thank Dr. Leo Carlin for seeding the initial idea for the project as well as providing all the image sequences used to train and evaluate the system. I would also like to thank him for carefully reviewing the training datasets.

Third, I give my thanks to my friend Diego Aranda Pérez for his help in annotating the training image sequences as well as helping me stay focused, especially in the initial weeks of the project.

Fourth, my friend Stefan McCarthy for proofreading this report.

Fifth, Dr. Carlos Arteta for making publicly available the original code from his research. His cell detection code has been reused in this project and allowed me to focus on the development of the cell tracking module.

Most of all, I give my thanks to my parents and brothers, for their guidance and support during the first two decades of my life.

The copyright of this thesis rests with the author and is made available under a Creative Commons Attribution Non-Commercial No Derivatives licence. Researchers are free to copy, distribute or transmit the thesis on the condition that they attribute it, that they do not use it for commercial purposes and that they do not alter, transform or build upon it. For any reuse or redistribution, researchers must make clear to others the licence terms of this work.



Contents

1	Introduction	DRAFT I	8
1.1	Motivation	DRAFT I	8
1.2	Objectives	DRAFT I	9
1.3	Contributions	DRAFT I	9
1.4	Report structure	DRAFT I	10
2	Related work	DRAFT I	11
2.1	Cell detection	DRAFT I	11
2.1.1	Cell segmentation using the Watershed technique	DRAFT I	11
2.1.2	Cell segmentation using level sets	DRAFT I	12
2.1.3	Cell detection by model learning	DRAFT I	12
2.2	Cell tracking	DRAFT I	13
2.2.1	Tracking by model evolution	DRAFT I	13
2.2.2	Tracking by frame-by-frame data association	DRAFT I	14
2.2.3	Tracking with a dynamics filter	DRAFT I	14
2.2.4	Cell tracking by global data association	DRAFT I	15
2.3	Conclusion	DRAFT I	16
3	Detection of cells	DRAFT I	17
3.1	Method overview	DRAFT I	17
3.2	Detection of candidate regions	DRAFT I	18
3.3	Inference under the non-overlap constraint	DRAFT I	19
3.4	Learning the classifier	DRAFT I	20
3.5	Feature selection	DRAFT I	20
3.6	Performance improvements	DRAFT I	22
4	Tracking of cells	DRAFT I	23
4.1	Method overview	DRAFT I	23
4.2	Joining cell detections into robust tracklets	DRAFT I	25
4.3	Global data association	DRAFT I	27
4.4	Implementation using linear programming	DRAFT I	28
4.5	Hypotheses likelihood definitions	DRAFT I	29
4.6	Computing the likelihoods	DRAFT I	30
4.7	Features for the linking classifier	OUTLINE	33
4.7.1	Estimating the velocity with Kalman filters	NEW	34
4.7.2	Gaussian broadening feature	DRAFT I	34
4.7.3	Best feature selection	NEW	34

4.8	Implementation details	NEW	35
5	Data acquisition and annotation	DRAFT I	39
5.1	Data acquisition and example datasets	DRAFT II	39
5.1.1	Datasets	DRAFT II	40
5.1.2	Image analysis challenges	DRAFT II	43
5.1.3	Manual data annotation	DRAFT I	45
5.2	The annotation tool	DRAFT I	45
6	Experimental results	DRAFT I	48
6.1	Cell detector	DRAFT I	48
6.1.1	Performance metrics	DRAFT I	49
6.1.2	Detection accuracy	DRAFT I	49
6.1.3	Computations time	DRAFT I	53
6.2	Cell tracker	DRAFT I	53
6.2.1	Performance metrics	DRAFT I	55
6.2.2	Tracking accuracy	DRAFT I	56
6.2.3	Computation time	DRAFT I	59
6.3	Limitations and areas of improvement	DRAFT I	60
6.4	Summary	DRAFT I	61
7	Conclusions and future work	DRAFT I	62
7.1	Conclusion	DRAFT I	62
7.2	Future work	DRAFT I	63
	Appendices		65
	A User Guide for the Annotation Tool		66
	B User Guide for the Interactive Annotation Viewer		67
	C Cell detection results		68
	D Cell tracking results		72
	Bibliography		75

Todo list

Figure: Examples of MSER detections	19
Figure: Tracking terminology	23
■ Make sure to update this if no longer iterative	24
Figure: Tracking process flow diagram	24
Figure: Robust tracklets	25
■ To finish this section I need to train a model with more data and decide whether to use NB or ANN. Then I need to describe the selected method.	26
■ Add a diagram to illustrate how the descriptors of annotated cells where found, because it's hard to understand from the description	27
■ Add an example H, L and I for illustration	29
■ Finalize the description of the ANN when I settle on a final shape once I train it using all the data	31
■ Make sure to update this if i decide to build a classifier for each gap length	32
■ Try looking back only the number of frames you are trying to close... no further, since those could be linked later.	32
■ Augment this list as more features are added	34
■ This feature has not yet been implemented	34
■ This feature has not yet been implemented	34
■ Define how the best features have been selected	34
■ Describe which features performed best	35
■ This feature has not yet been implemented	35
■ This is series30green	40
■ This is series30red	41
■ This is series13greencropped	41
■ This is series14croppedclean	42
■ This is seriesm170_13cropped	43
■ Add a comment of how effective (or not) the traker is.	63
■ I need one concluding paragraph	63
■ Rewrite: Make sure to update this section after running the experiments. Explain how good or bad the methods is, which are the strongest and which the weakest points. if bad, try to rationalize why.	63

1 Introduction DRAFT I

In this introductory chapter we describe the problem of tracking cells in hundreds of microscopy images manually and motivate the development of an automatic cell tracker. We also outline the objectives and contributions of the project and provide a report outline for the following chapters.

1.1 Motivation DRAFT I

Recent advances in intravital microscopy enable us to study the behaviour of different cells without excessively modifying the natural environment in which these cells are found within the observed organism.

Neutrophils are a type of leukocyte that have a crucial role in the clearance of infections [1]. A significant reduction in the number of neutrophils in the human body (or in mice) leads to severe immunodeficiency or death. They can be found in the bone marrow, liver, spleen, lung and throughout the circulatory system. Direct observation of neutrophils should help explain their function in these organs.

Of special interest is the transit of neutrophils in the fine and specialized capillary network located in the lung. To transit, the neutrophils must come in contact with the endothelium of these small vessels. In order to protect against pathogens, neutrophils are potently cytotoxic. Therefore, their regulation in the lung is important as the consequences of their unwarranted activation would be severe.

A recent study [2] has shown that the lifespan of neutrophils is much longer than previously predicted (up to 12.5 hours for mice and 5.4 days for humans). Although this specific study is a source of doubtful criticism [3], the longevity of neutrophils increases during inflammation. This longer lifespan may permit them to perform a wider range of complex activities, beyond the clearance of infections. An analysis of their behaviour as observed through intravital microscopy could help reveal more of their roles.

Finally, there is accumulating evidence to support the existence of different lineages of neutrophils with discrete roles. Automated analysis of microscopic images could explain whether these are truly distinct lineages or if they instead all develop from the same neutrophil predecessor.

The observation of neutrophils requires the analysis of hundreds of frames of microscopy image sequences. The manual annotation of these image sequences in order to extract trajectories of

movement of neutrophils is a time consuming and error prone process. Manual annotation severely limits the amount of data that can be analysed and slows down the advancement of cell research. It would be a major advance to be able to rely on the automatic identification and tracking of cells over time in sometimes noisy complex images.

1.2 Objectives DRAFT I

The clarity of neutrophil image sequences obtained *in vivo* can vary considerably due to the combination of the raster scanning imaging technique and, in the case of the lung, the motion of the tissue induced by a mechanical ventilator. In the images we can observe artefacts such as out-of-focus frames and blurred images due to the jiggling tissue, etc. This can sometimes make it difficult to identify and segment individual cells.

The aim of this research is to develop methods for cell detection and tracking. This system should be able to accept image sequences of cells in tissue, identify the cells, and track their position over the entire sequence.

The identification of cells should take into account the nature of the imaging technique and the quality of the obtained images. Simple methods such as thresholding would be too unreliable; more advanced methods need to be investigated.

Tracking of cells should also take into account the nature of input data. Basic frame-by-frame tracking of cells is likely to have poor results because the cells frequently disappear into the depths of tissue or lose focus. Methods that take into account the temporal behaviour of cells are likely to result in more robust tracking.

1.3 Contributions DRAFT I

The work in this project led to a development of a robust pipeline for automatic cell tracking. The pipeline combines a cell detection algorithm from [4] and, to the best knowledge of the author, a novel tracking method that is directly based on the observed data.

The cell tracking module uses a global data association approach to reliably generate cell detection trajectories based on a global decision. This research has upgraded previous methods allowing them to rely on a large set of features obtained from observed data. A machine learning approach is used to compute likelihoods for linking cell detections into increasingly longer trajectories. Because the likelihoods are obtained directly from training examples, the method is able to produce good results even from noisy datasets, where several frames in a sequence can become out-of-focus, and in which cells can disappear and reappear over time, etc.

Finally, an image annotation tool is developed that allows simple dot annotation of cells and linking of cells among frames to represent trajectories. This tool can be used for generating the necessary

training data and further allows to visualise and inspect the results produced by the automatic cell tracking method.

1.4 Report structure DRAFT I

The rest of the thesis is structured as follows:

Chapter 2 consists of brief literature survey outlining existing methods for cell detection and tracking. The chapter gives a background of how the methods for cell detection and tracking have evolved over time.

Chapter 3 describes in more depth the cell detector from Arteta *et al.* [4] and outlines the modification that improved the detection speed.

Chapter 4 describes in detail the cell tracking module. It outlines the two step process of generating robust tracklets and then joining them into long trajectories using trained classifiers.

Chapter 5 represents an overview of example datasets upon which the tracking tool is tested and presents a cell annotation tool developed to ease the annotation of image sequences.

Chapter 6 evaluates the performance of the cell detection and tracking module to quantify their performance. It describes the experiments and also provides qualitative analysis of the results.

Chapter 7 includes some concluding remarks and ideas that could be implemented to continue the advancement in the field of automatic cell detection and tracking.

2 Related work DRAFT I

This chapter consists a survey of current methods for cell detection and tracking. It summarizes the different techniques in a structured way, and discusses the advantages and disadvantages of each method. The chapter is divided into three sections. Section 2.1 describes methods to perform cell detection on images containing cells. Section 2.2 presents methods used to track cells in a sequence of images. Finally, in section 2.3 we describe which methods seem most promising for our application of cell tracking.

2.1 Cell detection DRAFT I

Cell detection consists of identifying individual cells in microscopy images. Due to the different imaging techniques and the types of cells we may wish to detect, several algorithms have been developed, each to handle a specific case. This section is an overview of these techniques.

2.1.1 Cell segmentation using the Watershed technique DRAFT I

A basic cell detection method relies on binarizing an image to separate the background from the cells, followed by a segmentation step to extract the cells. Chen, Biddell *et al.* [5] use a flooding approach for segmentation. The entire method can be summarized as follows. First, a spatial adapter filter eliminates some noise in the image. Second, it locates the pixels with minimum intensities in a small sliding window. Third, for each minimum point it proceeds to the progressive flooding of its neighbouring points and a final post-processing step discards false regions.

A similar method by Chen *et al.* [6] consists on using a Watershed algorithm [7] to separate cell nuclei after using Otsu's thresholding method to segment nuclei from the background. Morphological operations [8] can be used to fill holes and eliminate patches that are too small to correspond to cells. Additionally they developed a nuclei-fragment merging method based on the shapes and sizes of the nuclei to deal with the problem of over-segmentation caused by the Watershed algorithm.

Disadvantages of these methods include the requirement for a similar number of pixels belonging to both the cells and background as well as the high signal to noise ratio.

2.1.2 Cell segmentation using level sets DRAFT I

Another interesting technique to segment cells is a contour evolution method that makes use of the general appearance of cells to segment them using level sets. Mukherjee *et al.* [9] make the observation that leukocyte shapes are nearly circular and that at least for a significant part of the border of the cells, the intensity profile is different from the cell cytoplasm as well as the background. Using this observation, identification of a leukocyte is formulated as a minimization of an energy function incorporating image gradient and intensity homogeneity within the closed contour encompassing the cell. The benefit of this method is that it can be adjusted to perform well in images with significant clutter and poor contrast by increasing the importance of the homogeneity, or for images with good contrast, where the gradient magnitude term is given more importance. The disadvantage of this method is that cells cannot overlap. The energy function can be minimized with the gradient descent method.

To reduce the solution space for the energy function, only the boundaries of connected components within the image-levels sets are evaluated with the energy function. Only the connected components satisfying the size and shape constraints of the cells are extracted. The remaining components are eliminated using area morphology operations. This level-set analysis provides a more efficient solution that is linear in the number of intensity levels in the image in contrast to the much higher complexity of a curve evolution method.

The level set method is a contour evolution approach which has good results in segmentation. Tang *et al.* [10] have successfully combined level-sets and local grey thresholding [11] for segmenting neuron stem cells in images which have been obtained by confocal microscopy.

2.1.3 Cell detection by model learning DRAFT I

The previous methods perform efficiently in cells with sufficiently good contrast. In images where the cell borders are unclear, images are of varying intensity, cell density is high, or cells represent different shapes, these methods may not perform as well. In such cases machine learning methods can perform better by learning a model of a cell based on a training set of annotated examples.

Arteta *et al.* [4, 12], propose an algorithm that uses a highly-efficient MSER region detector [13] to find a broad number of candidate regions. These regions are then scored depending on the similarity to the cell type of interest by a machine learning algorithm.

The authors organized the extremal regions obtained from the MSER detector into trees, so that each tree corresponds to a set of overlapping extremal regions. Each region is given a value corresponding to the similarity of the region to a cell. The non-overlapping regions which achieve high scores can then be selected via dynamic programming of the trees. The learning is performed using a structured SVM [14] which is able to take into account the non-overlap constraint, and achieves good performance. The learning is performed on weakly annotated images – a single dot is necessary on each cell.

The advantage of this approach is the tolerance to changes in image intensities, cell densities and sizes. The major downside is the non-overlap constraint. Fortunately the authors have also developed an algorithm to detect partially overlapping cells [12].

The idea is to learn to detect overlapping cells together with the number of cells in the region. The algorithm starts by generating a set of nested regions. Each region is then scored using a set of classifiers that evaluate the similarity of the region to each of the possible classes, where each class corresponds to the number of cells that the region contains. An inference procedure then selects the non-overlapping subset of regions, and assigns each a class label indicating the number of cells that the model believes lie in the region. This eliminates the non-overlap constraint, but requires a larger training set of annotated images to train the detector.

2.2 Cell tracking DRAFT I

After detecting each cell in each image of the sequence we need to determine which cells in an image correspond to which in the next image. This way we can generate tracks of the cells across the image sequence. We have identified four main approaches to tracking:

Tracking by model evolution Active contours or level sets are used to detect a cell in the first frame of the sequence, and the cell models are then propagated to the next frame.

Tracking by data association Cells in consecutive frames are associated according to a similarity measure, which compares features such as the cell intensity and the relative spatial location.

Tracking with a dynamics filter This method uses a dynamics filter (e.g. Kalman or Particle filter) to predict the spatial location of the cell in the following frames.

Tracking by global data association This method approaches tracking as a global optimization problem and all trajectories are generated at once. These methods aim to maximize the probabilities that the generated trajectories are correct.

The remaining of this chapter briefly describes how these methods have been implemented in different papers.

2.2.1 Tracking by model evolution DRAFT I

This approach consists of identifying a cell in the first frame of a sequence by means of active contours or level sets and then propagating the models to the next frame. Model evolution tracking methods handle cell deformation well, but can often get stuck in local minima. This method can be computationally expensive because of the need to evolve the cell models for each frame. It is possible to improve the tracking accuracy at the expense of computation time.

Mukherjee *et al.* [9] model the problem of cell tracking as maximizing a similarity measure between

level sets in consecutive frames. The method minimizes the energy associated with leukocyte boundary detection and then matches the boundary with that of the previous frame. To maintain spatial coherence, the authors assume that the displacement of cells between frames is marginal. Such an automatic tracker is able to handle slight rotations or deformations of the leukocytes well.

2.2.2 Tracking by frame-by-frame data association DRAFT I

When tracking by frame-by-frame data association, cells in consecutive frames are associated according to a similarity measure, which compares features such as the cell intensity or relative spatial location. This method is very efficient, but only effective when cells are accurately detected. It is possible to improve the association by analysing several frames in a sequence.

Chen *et al.* [6] perform the matching process based on the distances between the cells. A match is found if a feature distance is below a threshold. The authors have also developed strategies to overcome the problem of false matches and ambiguous correspondence when nuclei are touching or partially overlapping. In this case information about these nuclei is stored (nuclei size and their relative location – up, down, left, right –). When they separate, the algorithm can resolve the ambiguities by comparing their current status with the previously stored information.

To reduce the vulnerability of frame-by-frame data association it is possible to analyse several frames of the sequence. This is the approach by Huh [16]. For each new frame, a set of hypothesis is computed for each detected blob, corresponding to migration, exit, entrance, clustering and mitosis. After all hypothesis are obtained, a best combination is selected by formulating and solving an integer programming problem. If, after the best association is found, there are remaining cells, these are considered again in the following frames.

The similarity function used to perform data association between consecutive frames is of crucial importance. House *et al.* [17] propose a novel cost function that encodes the response of cells to conditions of their environment. The tracking problem is formulated as a standard Bayesian filtering problem. The model allows the state of the cells in the images to evolve in time. To solve the complete assignment they have used a probabilistic data association algorithm which produces an optimal solution.

2.2.3 Tracking with a dynamics filter DRAFT I

Dynamics filters are a powerful addition to tracking systems because they predict the motion of a cell, and reduce the search space in which we look for matches. These algorithms not only perform frame-by-frame tracking, but also take temporal information into account, resulting in models that are robust to long-term occlusion and cell detection error.

Xu *et al.* [18] developed an ant stochastic searching behaviour based tracking system to track multiple cells in fluorescent image sequences. The system is similar to particle filters, but the motion behaviour is modelled to be similar to how ants behave when searching for food. When ants move

around, they deposit small quantities of chemical pheromone. Once they find food, they retrace the steps depositing larger amounts of pheromone. Other ants can then use this guide to find food faster, but are allowed to deviate from the path and pursue their own paths. The author proposes a system where the food is not static, so that it can be used for cell tracking. Compared to particle filters, this algorithm is more computationally expensive, but achieves better accuracy. The algorithm is able to deal with cells entering or leaving the scene as well as occlusion.

Kalman filters are another popular method for state prediction. Tang *et al.* [10] use a Kalman filter to guide the tracking of active cells. Active cells are those that move a distance larger than a threshold (supposedly their radius) between frames. Inactive cells are tracked separately by observing their overlap region.

Cells do not always exhibit just one type of motion and a single Kalman filter cannot take this into account. Li *et al.* [19] propose an automated tracking system which runs several Kalman filters in parallel, each corresponding to a different modality: random walk, first-order and second-order linear extrapolations which correspond to Brownian motion, migration with constant speed and migration with constant acceleration. The Kalman filters are run by the interacting multiple models (IMM) filter. The transitions between models is determined by a finite state Markov chain. Their system is composed of five modules: cell detector, cell tracker, dynamics filter, track compiler and track linker. The cell detector separates the background from cell pixels. The cell tracker propagates the cell labelling to the next frame in the sequence. The track compiler compares the results of the cell tracker, cell detector and dynamics filter to create a new track for new or daughter cells, and updates or terminates existing tracks. The track linker connects two or more track segments if they belong to the same cell.

2.2.4 Cell tracking by global data association DRAFT I

The benefit of global data association approaches is that they aggregate the results of all frames and make a global decision rather than propagating the results of each frame to the next. This makes them less dependent on errors from the cell detection module.

Massoudi *et al.* [20] propose a tracking method that does not rely on perfect segmentation and can deal with uncertainties by exploiting temporal information and aggregating the results of many frames. The system only requires probabilities or potentials that represent cell positions. Tracking is modelled as a network flow problem and is formulated as a linear program based on the occupancy likelihood of the edges of the network. The system can detect false positives or false negatives and correct itself. The method handles division events and cells entering or exiting the screen at the boundaries.

Zhang *et al.* [21] also researched tracking of pedestrians using network flows and achieved superior performance to frame-by-frame methods. In their approach, data association is defined as a maximum-a-posteriori (MAP) estimation problem which is solved as a flow network. The flow paths in the network correspond to non-overlapping trajectory hypothesis, and the flow costs to the observation likelihoods and transition probabilities. The global association is found by a minimum cost flow

algorithm.

An equivalent MAP problem is defined by Huang *et al.* [22] and Bise *et al.* [23]. First, they both generate reliable tracklets by linking cell detection responses in consecutive frames. Second, the short tracklets are associated into longer and longer tracklets. This second, global data association approach is solved by Huang *et al.* iteratively using the Hungarian association algorithm [24] and by Bise *et al.* as an integer optimization problem. This approach can achieve state-of-the-art performance and surpass the accuracy of previously discussed methods.

2.3 Conclusion DRAFT I

This chapter presented an overview of methods used for cell detection and tracking. Some of these methods are designed to perform better under specific imaging and cell conditions. The combination of thresholding and the Watershed method for segmentation is likely to perform well in high quality images where the cells are clearly discernible. However, in the case of noisy images with varying contrast, the machine learning approach presented by Arteta *et al.* is likely to perform much better, at the cost of computation time to compute the larger number of features for each candidate cell region. Both these methods have been briefly evaluated on the noisy image sequences that we studied in this research. The first method required a lot of manual adjustments to perform acceptably well. The second method was able to reliably detect cells in the low signal-to-noise images. For this reason, the machine learning method by Arteta *et al.* has been chosen for this application. A more in depth explanation of the method is presented in chapter 3.

Although the detection method is able to reliably detect most of the cells in the image sequences, the cells sometimes cannot be detected due to several factors. First, the cells can submerge into the depth of the tissue and reappear a few frames later. Second, due to the motion artefacts caused by the moving (lung) tissue when the images are obtained, the images can be out-of-focus for several frames. These and other artefacts make the cell tracking problem much harder. Therefore, we needed a method that would be able to handle false detections and missing observations over a few frames. A global data association method heavily inspired by [23] has been developed and is presented in chapter 4.

There are several reasons why automatic cell tracking systems are not as popular as we would expect. The main reason is probably that the level of accuracy falls short of manual annotation. Furthermore, these systems are often not robust to changes in cell type, cell culture condition and microscopy image quality. It is our hope that the state-of-the-art methods we have chosen and will develop for our tracking application will enable us to track cells in noisy images with a performance comparable to that of a human.

3 Detection of cells DRAFT I

This chapter presents an overview of the cell detector from [4]. The method has been chosen for its ability to learn to detect cells in the low contrast images we aim to analyse with the tracking pipeline.

3.1 Method overview DRAFT I

In order to track cells across frames we need to first be able to identify the cells within the images. As described in section 2.1, there is a broad range of automatic methods for cell detection and segmentation.

The requirements for detecting cells in our application are:

1. Accurate detection in sometimes noisy images. The detector should be able to robustly identify cells in datasets that include noise, motion artefacts (induced by the camera and the moving tissue) and variable lens focus. The detector does not need to detect cells that are out-of-focus, but should reliably detect those which are in focus. Missed detections are dealt with in the tracking module.
2. There is no need for accurate segmentation. We are primarily interested in the accurate positional tracking of cells over time, and analysis of the cells appearance is of secondary interest. For the purpose of tracking, it is sufficient to be able to reliably localize cells, and extract some features that quantify them.
3. The algorithm should adapt well to different microscopy modalities. The studied datasets are obtained from various organs within the body of mice that have distinct textures. The detector should be able to detect cells in any of these datasets with minimal manual adjustments.

Much of the previous work described in section 2.1 was focused on an accurate segmentation of cells. Many of the described methods would return very good segmentations, but they would fail when presented with a very noisy dataset of varying contrasts, or need major manual tuning of parameters when presented with a new dataset with a different kind of cell.

For the purpose of our application we have chosen the cell detector presented in [4], because it conforms to all three requirements. The machine learning method is able to handle noisy datasets, and the model can easily be retrained when a new, previously unseen, type of image needs to be

analysed. The method does not focus on an accurate segmentation of cells, but on the robust localization. The major drawback of this method was performance. The authors reported run-times of 30 seconds to detect cells on 400-by-400 pixel images on an i7 CPU. This slow performance would make it unusable for tracking large image sequences. However, we were able to optimize the MATLAB code for feature computation so that the detection process took only a fraction of the reported run-time.

The detection process runs in three steps. First, robust candidate regions are extracted from an image. The method uses an efficient MSER region detector [25] to find a large number of candidate regions. Second, each of these regions is assigned a value which is produced by a classifier; this is a score of similarity between the region and a true cell. Finally, the non-overlapping subset of these candidate regions with high similarity to the annotated cells in the training data are selected via dynamic programming.

The detector can be trained on a small number of training images, where each cell is annotated with a single dot.

The code used in our method is based on the original code from [4]. Some of our modifications, especially in the feature computations, allow the detector to run significantly faster. Additional work was required to combine the cell detector with the tracking module described in chapter 4.

The rest of this chapter is divided as follows. In section 3.2 we describe how the candidate regions are obtained. Section 3.3 shows the inference procedures that selects the optimal subsets of candidate regions. In section 3.4 we formulate the learning method and in section 3.5 we describe the choice of features. Finally, in section 3.6 we briefly outline the main changes that improved the performance of the original algorithm.

3.2 Detection of candidate regions DRAFT I

The first stage of the cell detection process is the extraction of a large number of candidate cell regions. A region is extremal if the image intensity everywhere inside of it is higher than the image intensity at its outer boundary. For a grayscale image I it is the set of connected components of the thresholded image $\mathcal{I}_{>t} = \{I > t\}$ for some threshold t . In practice we consider only regions that are maximally stable, as defined in [25], obtained using an efficient maximally stable region detector (MSER).

Extremal regions are invariant to affine transformations of image intensities. They are stable, which means that their support is virtually unchanged over a range of thresholds, and they can be detected at multiple scales and be enumerated very efficiently. This makes them ideal for detecting candidate regions that correspond to cells in microscopy imaging. Figure 3.1 shows an example result of running the MSER detector on a sample set of microscopy cell images.

An important property of extremal regions is *nestedness*, which means that two extremal regions detected on the same image can be either nested or non-overlapping.

The number of extracted candidate regions is high, and an inference step determines which belong to cells.

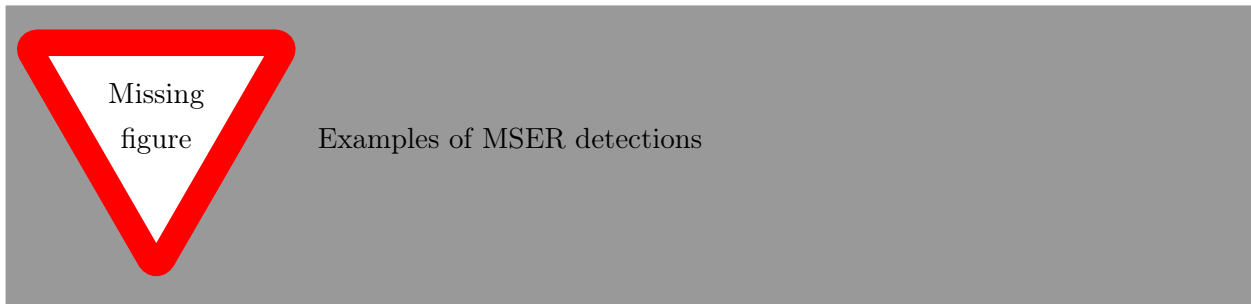


Figure 3.1: MSER detection examples

3.3 Inference under the non-overlap constraint DRAFT I

The inference procedure selects a subset of candidate extremal regions that correspond to cells. Let V_i be a value assigned to each candidate region R_i indicating the appropriateness of the region to correspond to a cell. The inference procedure selects a subset of the non-overlapping candidate regions such that the sum of the values of selected regions is maximized. To make the problem computationally feasible, the regions are organized into trees according to their nestedness property. Each tree represents a set of overlapping extremal regions. The problem can then be formalized as follows. Let $\mathbf{y} = \{y_1, y_2, \dots, y_N\}$ be a binary variable such that $y_i = 1$ if the region R_i is selected. Let \mathcal{Y} be the set of non-overlapping regions. The maximization problem can be defined as:

$$F(\mathbf{y}) = \max_{\mathbf{y} \in \mathcal{Y}} \sum_{i=1}^N y_i V_i.$$

Further details on how the exact solution is obtained can be found in [4].

Although cells in the studied datasets occasionally overlap, detecting only non-overlapping cells is a good balance between the manual work required to annotate datasets, the training time and achieving accurate detections. The described framework has been expanded by the original authors to allow for the detection of partially overlapping regions [12]. However, that model requires a longer training time, and a larger set of manually annotated images to accurately detect blobs of overlapping cells. Finally, the global data association technique used in the cell tracker module should be able to correctly associate cells if the overlap time is short, such as in trajectories that are crossing and then continuing in opposite directions.

3.4 Learning the classifier DRAFT I

The value V_i assigned to each region R_i is a classifier score that indicates the similarity of the extremal region to a cell. The classifier is trained using dot-annotated training images $\mathcal{I}^1, \mathcal{I}^2, \mathcal{I}^3, \dots$ as follows. Let $R_1^j, R_2^j, \dots, R_{N^j}^j$ indicate the set of N^j candidate regions extracted from the training image \mathcal{I}^j . The value V_i^j of each region is computed as the dot product between a feature vector \mathbf{f}_i^j computed for each region and a weight vector w : $V_i^j = \mathbf{f}_i^j \cdot w$.

The learning procedure has to learn the weight vector w so that the inference procedure tends to pick regions with a single annotated cell $n_i^j = 1$. In order to consider the non-overlap constraint of regions, a structured SVM [26] is used that directly optimizes the performance of the inference on the training set. The loss function that the learning framework tries to optimize counts the number of deviations from the one-to-one correspondence between the annotated dots and the selected regions:

$$L(\mathbf{y}^j) = \sum_{i=1}^{N^j} y_i^j |n_i^j - 1| + U^j(\mathbf{y}^j),$$

where n_i^j indicates the number of annotated dots in region R_i^j and $U^j(\mathbf{y}^j)$ counts the number of annotated dots that do not have a corresponding selected region R_i^j . Further details about the convex objective we tried to minimize and how it can be optimized using a cutting-plane algorithm can be found in [4].

3.5 Feature selection DRAFT I

The effectiveness of the machine learning method to detect cells depends on the quality of features computed for the regions. Good features have a lot of discriminative power, so that they can effectively distinguish regions corresponding to cells from regions that don't correspond to cells. In order to find the best feature vector, we have evaluated the performance of the classifier with several combinations of these features:

1. The area A of the region represented by a 10-dimensional binary vector with the entry $\lceil \log A \rceil$ set to 1.
2. 10-dimensional histogram of intensities within the region.
3. The position of the region in the image in terms of x-y coordinates of the region centroid.
4. Two 6-dimensional histograms of differences in intensities between the region border and a dilation of it for two different dilation radii.
5. A shape descriptor represented by a 60-dimensional histogram of the distribution of the boundary of the region on a size-normalized polar coordinate system.

6. The orientation of the descriptor after attempting to normalize its orientation.
7. The proportion of edge-pixels in the region.

Each of these features has different discriminative power, and takes some time to compute. The application of the cell detector requires that images are processed within a time limit. For this reason, we have trained and tested the algorithm with all the possible combinations of these features and identified a feature set that computes in an acceptable amount of time with a high recall and precision.

We have also developed a tool that helps select the most appropriate set of features given specific constraints, for example a maximum computation time, minimal precision and recall values, etc. A graph generated by the function is shown in figure 3.2.

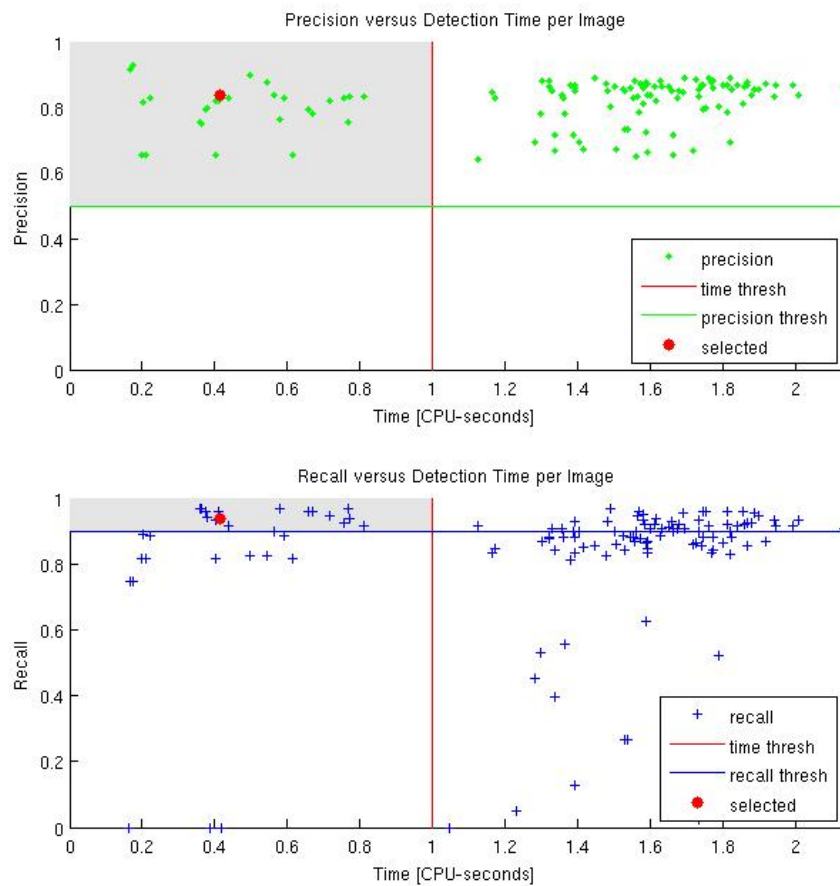


Figure 3.2: A function helps select an appropriate feature set according to constraints, such as the computation time per image, mean precision and recall. This example shows features sets that compute within 1 CPU-second per image on average (to the left of the red vertical line), have at least 0.5 precision (above the green horizontal line) and 0.9 recall (above the blue horizontal line). The feature set corresponding to the selected parameters shown as red dots contains features 1, 3 and 4. Most importance was given to high precision followed by low computation time and high recall. The selected feature set computes in about 0.4 CPU-seconds per 512-by-512 pixel image with mean precision of 0.836 and mean recall of 0.9363 (as measured on an i7-2600 CPU with a clock frequency of 3.40 GHz).

3.6 Performance improvements DRAFT I

The main drawback of the algorithm presented by Arteta [4] is the slow feature computation. The original paper reported detection times of 30 seconds for images of dimensions 400-by-400 pixels on an i7 CPU. Because the aim of this project is tracking cells, we expect to be processing hundreds or thousands of frames of microscopy image sequences. It was important to reduce the detection time as much as possible. The major performance improvements were achieved by addressing three things.

The algorithm computes the features for the classifier on every candidate region. The original method was configured such that thousands of MSERs were computed for each frame. While this provides more robust learning and detection, it was slowing the algorithm down excessively. First, we fine-tuned the MSER detector to detect less regions, but still return a large number of features such that the detection rate wasn't penalized.

Second, we have identified the slowest computed features and improved the algorithmic behaviour of the original MATLAB implementation. One such feature is the Contour Points Distribution Histogram. The function was frequently calling slow, built-in functions, to extract region characteristics. This, and other functions, were rewritten to call these expensive sub-procedures less frequently, without affecting the behaviour of the algorithms.

Third, we noticed that the MATLAB built-in functions were performing a lot of parameter checking, to make sure that the input parameters were valid. Many of these parameters were strings, and string manipulations are very slow in MATLAB. Several MATLAB built-in functions were rewritten or replaced to remove excessive parameter checking, which in several cases represented an overhead of over 30%. These parameter checks are welcome when developing the algorithm, but once the algorithm is complete, several of these checks can be safely removed.

It is also worth mentioning that the updated original code for the detector is now more stable and better tested. A few rare bugs have been fixed, after being found when running the detector over a large number of datasets.

These optimizations resulted in a significant performance boost. The updated algorithm can perform the same computations in a fraction of the time taken by the original implementation provided by the authors of [4].

4 Tracking of cells DRAFT I

This chapter describes the method for tracking cell detection results to obtain cell trajectories. The chapter is a step-by-step description of the process, starting in section 4.1 with a high level overview of the tracking process. The following chapters describe each step in more detail. Section 4.2 explains how cell detection results are joined into short robust tracklets. In section 4.3 we formulate a maximum-a-posteriori problem to link these short tracklets into robust trajectories, and then in section 4.4 the problem is converted to an integer optimization problem and solved by linear programming. The final three sections describe specific details of the linear programming solution. In section 4.5 we define the set of hypothesis (false positive, linking, tracklet initialization and tracklet termination) for each tracklet and in section 4.6 we describe how the likelihoods of these hypothesis are computed. Finally, section 4.7 describes the different features that were used to train a classifier for linking trajectories.

4.1 Method overview DRAFT I

The cell detections obtained using the method described in chapter 3 need to be linked into trajectories. The detections contain a number of false positives and false negatives (missed detections) which make the task of associating them into trajectories difficult.

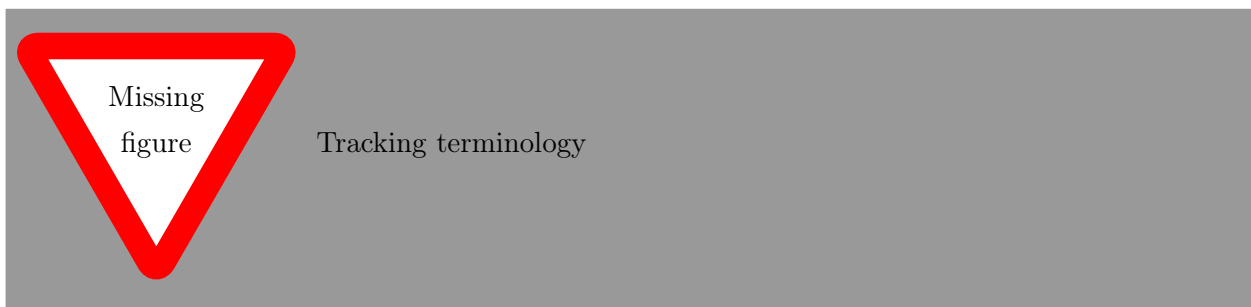


Figure 4.1: Tracking terminology

First, we define the terminology used in the subsequent sections. A robust tracklet is a sequence of cell detections that can be linked with high confidence. These are likely to be detections that were segmented with high accuracy and their feature vectors are very similar. A robust tracklet cannot have gaps (missing detections). Similarly a tracklet is a sequence of cell detections, but differ from robust tracklets in the manner in which it may contain gaps (missed detections). A sequence of one or more robust tracklets is a tracklet, but the opposite is not necessarily true. Finally, we call a trajectory a sequence of tracklets that cannot be effectively linked to any other tracklets. A

trajectory is a maximally linked tracklet, and corresponds to the actual path performed by a cell in the image sequence. These concepts are illustrated in fig. 4.1.

Linking cell detections into tracklets is performed in two steps, as seen in fig. 4.2. First, cell detections are linked into robust tracklets by a reliable linking model. Second, the tracklets are iteratively

Make sure to update this if no longer iterative

associated into ever longer tracklets by closing ever larger gaps. A detailed overview of these two steps is provided in the following sections.

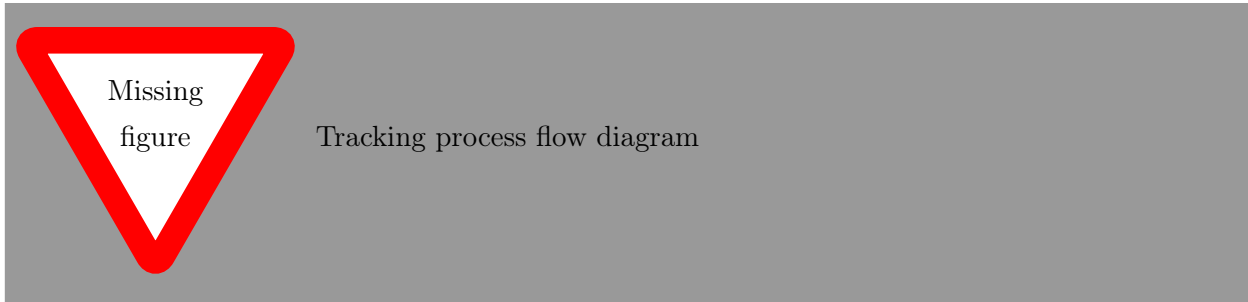


Figure 4.2: Some caption

The process of linking robust tracklets into trajectories is performed globally, by selecting the optimal subset of tracklets to link in each iteration. This global data association approach has been chosen because it has shown significant improvements in tracking performance compared to other successful methods [23], such as [27] which performs tracking using an Interacting Multiple Models (IMM) filter that runs several Kalman filters in parallel for each trajectory.

The approach used in this project is similar to [15] in the way in which it formulates the maximum-a-posteriori probability (MAP) problem and solves it using linear programming, but the likelihoods of the hypothesis (the likelihood of linking two tracklets, the likelihood of a tracklet being the first tracklet in a trajectory, the likelihood of it being the last tracklet in a trajectory or the likelihood of it being a false positive) were computed using a machine learning approach. This approach makes it possible to associate cells in very noisy and low quality images, where accurate cell detections are not always possible.

Although the developed system is fully automatic, it gives some control to the user by letting him adjust the hypothesis' likelihoods by means of four intuitive parameters and thus change the way the tracklets are linked. This control eliminates the need to retrain a specific model for each new dataset, as it makes it possible to re-use a trained linking classifier to link robust tracklets in an independent dataset and adjust these parameters in order to obtain an adequate set of cell trajectories.

Finally, let's examine the importance of recall and precision values from the cell detection module and how these affect the process of generating cell trajectories. A high precision value indicates a small number of false positive detections. The cell tracker module is able to effectively eliminate these false positives. However if the precision value is low, then these false positives could start forming structures that the cell tracker could confuse for actual cell trajectories. It is therefore

important that the cell detection module should not detect a too large number of false positives. A high recall value indicates that most of the manually annotated cells have been detected, which means that the cell detections will be easily linked to form trajectories as seen in fig. 4.3. The tracker will work best with very high recall values. If the recall value is low, there will be large gaps between individual detections. The cell tracker will then have to link short tracklets that can be further apart, which is likely to increase the number of incorrect links.

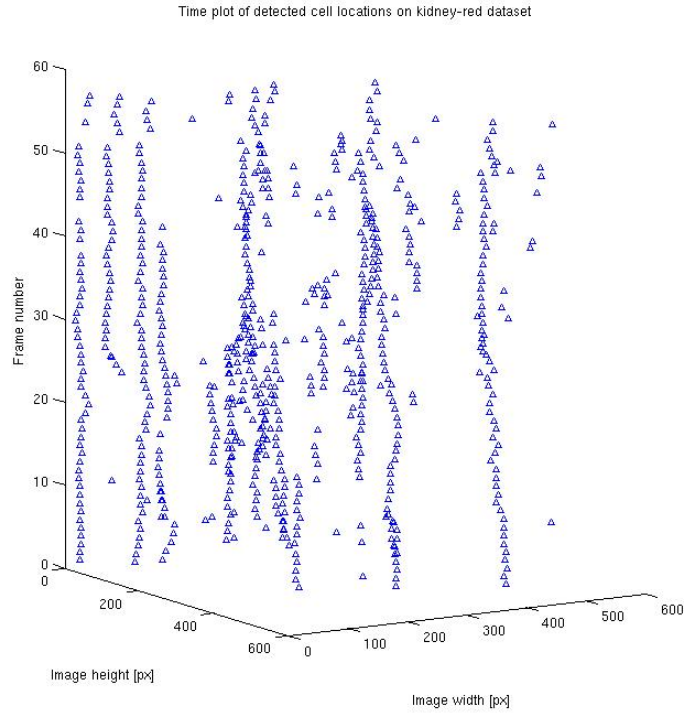


Figure 4.3: Cells detected over 60 consecutive frames are visualized as a time series. The vertical axis corresponds to the frames. Even in this raw detection data, it is possible to see the tracks of some of these cells.

4.2 Joining cell detections into robust tracklets DRAFT I

The first phase of linking cell detections into trajectories consists of identifying a set of robust tracklets. An example dataset with the identified robust tracklets is shown in fig. 4.4



Figure 4.4: Robust tracklets

We define a cell detection $d_i = (x_i, y_i, f_i, t_i)$, where x_i and y_i are the position of the cell detections

within the frame, f_i an appearance feature vector obtained from the cell detector module, and t_i the frame index of the detection. The set of all cell detections in the image sequence is $\mathbf{D} = \{d_i\}$.

Let $\mathbf{T} = \{T_k\}$ be a hypothesis set of tracklets, where each tracklet is defined as a list of robust tracklets, such that the frame index of the last detection ($t_{k_{i_n}}$) in each robust tracklet is lower than the frame index of the first detection of any following tracklet ($t_{k_{i+1_1}}$): $T_k = \{T_k^{robust} | \forall i, t_{k_{i_n}} < t_{k_{i+1_1}}\}$. A robust tracklet is defined as $T_k^{robust} = \{d_{k_i} | \forall i, t_{k_{i+1}} = t_{k_i} + 1, d_{k_i} \in \mathbf{D}\}$. Assuming that each detection can belong to only one tracklet we define the non-overlap constraint:

$$\forall T_i, T_j \in \mathbf{T}, i \neq j, T_i \cap T_j = \emptyset.$$

For each cell detection we need to identify a good match in the next frame, if it exists. Let P_{link} represent the likelihood of linking two detections d_i and d_j :

$$P_{link}(d_j | d_i) = \begin{cases} V((f_i, x_i, y_i), (f_j, x_j, y_j)) & \text{if } t_j - t_i = 1 \\ 0 & \text{otherwise} \end{cases},$$

where V is an affinity function that returns the probability that the provided feature vectors and detection positions belong to the same cell. The feature vectors are obtained from the cell detection module and correspond with the feature vectors used to identify a candidate region as a cell. This avoids the need to compute new feature vectors, which could significantly slow down the tracking process.

The probability of linking detections P_{link} is computed on all pairs of cells between consecutive frames i and $i + 1$ and vice versa. For each cell detection we found the most similar detection in the next frame, and for each detection in the second frame the most similar detection in the previous frame. Then, only symmetrical matches were chosen. A matching is symmetric if a detection in the first frame d_1 best matches detection d_2 in the second frame, and detection d_2 best matches detection d_1 in the first frame. This way we obtain a subset of matching pairs, such that each detection is matched to exactly one or no detection in the next frame.

To increase the robustness of the matches, a threshold θ_1 was chosen, such that only cell detection pairs whose linking probability was higher than that threshold could be linked.

The affinity function V is learned using a supervised machine learning algorithm, which learns to solve the binary classification problem of linking (or not linking) a pair of cell descriptors. The model is learned by comparing the appearance feature vectors of the detections as well as their positions, which are obtained from the cell detection module.

To finish this section I need to train a model with more data and decide whether to use NB or ANN. Then I need to describe the selected method.

The machine learning algorithm was trained with annotated datasets. The annotations contained

the position of cells in the images and links to matching cells in consecutive images. To train a linking model, cell appearance feature vectors had to be obtained for the dot-annotated cells. First, the cell detection module was trained to detect cells in a dataset. Second, the detector found a set of cell detections, which were matched to the corresponding real annotations. A detection was matched to an annotated cell if and only if the position difference was below a small threshold (10 pixels). This step was required to obtain the cell descriptors of the dot-annotated cells. Finally, the data to train the classifier was constructed as follows. Positive examples were selected as consecutive (linked) cells within each tracklet. Negative examples were chosen from all possible combinations of cells from different tracklets.

Add a diagram to illustrate how the descriptors of annotated cells were found, because it's hard to understand from the description

Based on the pairs of symmetric matches between cell detections, the system is able to generate a set of robust tracklets \mathbf{T}^{robust} . Cell detections that were not linked to any other cell are also considered robust tracklets, and are included in \mathbf{T}^{robust} for possible further linking in future steps of the algorithm.

4.3 Global data association DRAFT I

The problem of linking robust tracklets together to form trajectories is formulated as a MAP problem [21–23]. We first present the formulation, and then in the next section we provide the linear programming implementation.

Given the robust tracklet set \mathbf{T}^{robust} , we maximize the posteriori probability to find the best data association:

$$\begin{aligned}\mathbf{T}^* &= \arg \max_{\mathbf{T}} P(\mathbf{T} | \mathbf{T}^{robust}) \\ &= \arg \max_{\mathbf{T}} P(\mathbf{T}^{robust} | \mathbf{T}) P(\mathbf{T}).\end{aligned}$$

Assuming that the likelihood probabilities of the robust tracklets are conditionally independent given \mathbf{T} , and $T_k \in \mathbf{T}$ cannot overlap with each other, i.e. $\forall T_i, T_j \in \mathbf{T}, i \neq j, T_i \cap T_j = \emptyset$:

$$\mathbf{T}^* = \arg \max_{\mathbf{T}} \prod_{T_i \in \mathbf{T}^{robust}} P(T_i | \mathbf{T}) \prod_{T_k \in \mathbf{T}} P(T_k).$$

The likelihood of a robust tracklet is defined as:

$$P(T_i|\mathbf{T}) = \begin{cases} P_{TP}(T_i) & \text{if } \exists T_k \in \mathbf{T}, T_i \in T_k \\ P_{FP}(T_i) & \text{if } \forall T_k \in \mathbf{T}, T_i \notin T_k \end{cases},$$

where $P_{TP}(T_i)$ is the probability of T_i being a true positive and $P_{FP}(P_i)$ the probability of T_i being a false positive.

The probability of a tracklet $P(T_k)$ is modelled as a sequence of observations with the Markov property, namely that, given the current observation, the previous and future observations are independent:

$$\begin{aligned} P(T_k) &= P(\{T_{k_1}, T_{k_2}, T_{k_3}, \dots, T_{k_n}\}), \text{ where } T_{k_i} \in \mathbf{T}^{robust} \\ &= P_{init}(T_{k_1}) P_{link}(T_{k_2}|T_{k_1}) P_{link}(T_{k_3}|T_{k_2}) \dots P_{link}(T_{k_n}|T_{k_{n-1}}) P_{term}(T_{k_n}) \\ &= P_{init}(T_{k_1}) \left[\prod_{j=1:n-1} P_{link}(T_{k_{j+1}}|T_{k_j}) \right] P_{term}(T_{k_n}), \end{aligned}$$

where $P_{init}(T_{k_1})$ is the probability of T_{k_1} being the first robust tracklet in T_k , $P_{term}(T_{k_n})$ the probability of T_{k_n} being the last robust tracklet in the sequence, and $P_{link}(T_{k_{j+1}}|T_{k_j})$ transition or linking probabilities for $T_{k_{j+1}}$ and T_{k_j} . The definitions of these terms will be provided in section 4.6.

Note that the MAP problem takes into consideration the possibility of false cell detections, which makes the model ideal for very noisy and low quality microscopic image sequences where the cell detector is likely to find a number of false detections. Additionally, in the analysed image sequences there are often gaps of several frames where the image becomes out of focus and the cells disappear from the field of view. The linking probabilities permit an efficient closing of these gaps.

The benefit of the global data association approach to cell tracking is that it makes a global decision based on the probabilities defined over all the frames of the image sequence rather than propagating the results from frame to frame. This makes the algorithm more robust to errors in the cell detection module.

4.4 Implementation using linear programming DRAFT I

The MAP problem is converted to an integer optimization problem and solved by linear programming.

Let N be the number of input robust tracklets. Let L be a vector containing the likelihoods of all possible hypothesis: initialization, termination, false positive, and linking hypothesis between two robust tracklets. The formulation of these likelihoods is given in section 4.5, and the implementation details in section 4.6.

We also define a matrix H of dimensions $|L| \times 2N$ containing constraints to avoid selecting conflicting hypothesis. Let i represent the index of each new hypothesis and j the index over the columns of H . Then, for a robust tracklet T_k and candidate linking tracklet T_l the entries of matrix H are defined as follows for each possible hypothesis:

$$C_{ij} = \begin{cases} 1 & \text{for an initialization hypothesis if } j = N + k \\ 1 & \text{for a termination hypothesis if } j = k \\ 1 & \text{for a false positive hypothesis if } j = N + k \text{ or } j = k \\ 1 & \text{for a linking hypothesis if } j = k \text{ or } j = N + l \\ 0 & \text{otherwise.} \end{cases}$$

Once the constraint matrix H and likelihood vector L are defined, the original MAP problem from section 4.3 can be solved by selecting a subset of rows from H such that the sum of the corresponding likelihoods in L is maximized, under the non-overlap constraint of tracklets. The MAP problem can be reformulated as a binary linear problem:

$$I^* = \arg \max_I L^T I, \text{ subject to } H^T I = 1,$$

where I is a binary vector containing 1 for the selected rows of the matrix H and 0 elsewhere. The constraint $H^T I = 1$ guarantees that each robust tracklet appears in only one tracklet, or is discarded as a false positive.

For each tracklet an initialization, a termination and a false positive hypothesis is computed. The linking hypothesis is computed for pairs of tracklets where the gap between the tail of the first and head of the next tracklet is shorter than a user specified number of frames.

Add an example H , L and I for illustration

4.5 Hypotheses likelihood definitions DRAFT I

In this section we define how the different hypothesis are computed. In section 4.6 we will discuss the implementation details.

Due to errors in the cell detection module, all tracklets are candidates for being false positives. The false positive hypothesis likelihood is computed as:

$$L_i = \log P_{FP}(T_k).$$

The linking hypothesis measures the likelihood of connecting two tracklets. Candidate tracklet pairs

for linking are those for which the distance between the last detection (the tail) of the first tracklet (X_{k_n}) and the first detection (the head) on the second tracklet (X_{l_1}) is less than a specified number of frames. The likelihood of the linking hypothesis is computed as:

$$L_i = \log P_{link}(T_l|T_k) + \frac{\log P_{TP}(T_k) + \log P_{TP}(T_l)}{2}.$$

Bise *et al.* [23] dealt with cell tracking on image sequences where cell detections could be reliably detected. The authors considered tracklets close to the boundaries of the field of view as candidate for initial tracklets. This work is based on image sequences that were obtained from observing cells in thick tissue, where cells can sink into the background and reappear at any time. The initialization hypothesis indicates the likelihood that a tracklet is the first tracklet in a trajectory. Taking this into consideration new trajectories can be initialized anywhere within the field of view. The corresponding likelihood is computed as:

$$L_i = \log P_{init}(T_k) + \frac{\log P_{TP}(T_k)}{2}.$$

Similarly to the initialization hypothesis a cell can sink into the background or leave the field of view. This is taken into account in the termination hypothesis, which is also evaluated for all tracklets. The termination likelihood is computed based on the probability of the tracklet being at the end of a sequences:

$$L_i = \log P_{term}(T_k) + \frac{\log P_{TP}(T_k)}{2}.$$

A true positive tracklet appears in exactly two of the initialization, termination or linking hypotheses. For this reason the likelihood of a tracklet being true positive $\log P_{TP}(T_k)$ is divided into two halves that are included in these hypothesis.

4.6 Computing the likelihoods DRAFT I

In this section we describe the implementation details of the likelihoods: $P_{TP}(T_k)$, $P_{FP}(T_k)$, $P_{init}(T_k)$ and $P_{term}(T_k)$. The likelihoods are directly connected to the input observations and are estimated from training data.

The true and false positive likelihood DRAFT I

The true and false positive likelihoods of a tracklet T_k are defined in terms of the miss detection rate of the cell detector module α , and the number of the cell observations composing the tracklet which we denote $|T_k|$:

$$P_{FP}(T_k) = \pi_{FP} \alpha^{\frac{|T_k|}{\lambda_1}}$$

$$P_{TP}(T_k) = 1 - P_{FP}(T_k),$$

where λ_1 is empirically set to 2, and π_{FP} is a free parameter used to increase or decrease the likelihood that a tracklet is false positive.

The linking hypothesis DRAFT I

The linking hypothesis is computed between pairs of tracklets where the distance (number of frames) between the tail of the first and the head of the second tracklet is less than a threshold. Because of the variable contrast and poor quality of the imaging technique, the measure is computed by taking into account several spatio-temporal and visual features. The different features are outlined in section 4.7. The likelihood of linking tracklet T_l with T_k is:

$$P_{link}(T_l|T_k) = \begin{cases} \pi_{link} V(T_l, T_k) & \text{if } t_{l,1} - t_{k,n} \leq \lambda_2 \\ 0 & \text{otherwise,} \end{cases}$$

where $t_{k,n}$ is the frame index of the last detection of tracklet T_k , $t_{l,1}$ is the frame index of the first detection of tracklet T_l , λ_2 is a threshold indicating the maximum allowed gap for linking two tracklets, $V(.)$ a function that returns the probability that the tracklets should be linked, and π_{link} a free parameter used to scale the linking likelihood.

The function $V(.)$ is a model incorporating appearance and spatio-temporal features of the candidate linking tracklets. It is trained using an artificial neural network (ANN) with the number of inputs equal to the number of features and a single output that returns a value between 0 and 1.

Finalize the description of the ANN when I settle on a final shape once I train it using all the data

The binary ANN is trained using annotated image sequences. To generate positive training examples, the annotated trajectories were split into several parts. Then the combinations of these segments belonging to the same original trajectory were used as positive examples in the training data. Similarly, the combinations of these segments belonging to different original trajectories were used as negative examples. One of the features used in the classifier is the appearance vector, which is obtained for each cell annotation using the cell detection module.

The data used to train the classifier contains features of pairs of tracklets that can be separated by a different number of frames, from 1 to λ_2 . The consequence of using a binary classifier is that it might return a larger probability of linking two tracklets that are further apart than two tracklets that are closer together. For example, an original trajectory could be detected as three robust tracklets that should be linked sequentially. The classifier might return a larger likelihood of linking the first and third tracklet than the first and second tracklet. The data association module would

then likely link the first tracklet to the third, leaving the second one as a new short trajectory. To overcome this problem the process of linking tracklets is performed iteratively, closing ever larger gaps between tracklets. Each iteration takes the tracklets obtained in the previous one and allows for further association. This way the described problem can no longer occur. A positive side effect of performing the process iteratively is a reduced peak memory usage because of the lower number of hypothesis that are evaluated in each step.

Make sure to update this if i decide to build a classifier for each gap length

The benefits of this machine learning approach for computing the likelihood of linking two tracklets are twofold. First, it works well on noisy datasets because it uses a large number of features to train the model. Secondly, the large number of features makes it less vulnerable to inaccurate cell segmentation.

The initialization likelihood DRAFT I

The initialization likelihood is a measure of a tracklet being the first tracklet of a trajectory. This work is based on image sequences with high noise and contrast variance, where cell detections cannot be reliably detected over the entire trajectories of the cells. Additionally, the cells can sink into or surface from the background at any time. We take into consideration that new trajectories can be initialized anywhere within the field of view. The initialization hypothesis is based on the linking hypothesis. It is equal to the likelihood of tracklet T_k not being linked to the most likely linking tracklet in the λ_2 frames *before* its first cell detection:

$$P_{init}(T_k) = \begin{cases} \pi_{init}(1 - \max P_{link}(T_k|T_l)) & \forall T_l \in \mathbf{T}, t_{k,1} - t_{l,n} \leq \lambda_2 \\ 0 & \text{otherwise,} \end{cases}$$

where π_{init} is a free parameter that scales the initialization likelihood.

Try looking back only the number of frames you are trying to close... no further, since those could be linked later.

The termination hypothesis DRAFT I

The termination hypothesis measures the likelihood of a tracklet being the last tracklet of a trajectory. It is defined similarly to the initialization hypothesis: the likelihood of tracklet T_k not being linked to the most likely linking tracklet in the λ_2 frames *after* its last cell detection:

$$P_{term}(T_k) = \begin{cases} \pi_{term}(1 - \max P_{link}(T_l|T_k)) & \forall T_l \in \mathbf{T}, t_{l,1} - t_{k,n} \leq \lambda_2 \\ 0 & \text{otherwise,} \end{cases}$$

where π_{term} is a free parameter that scales the termination likelihood.

The scaling parameters DRAFT I

Each of the hypothesis likelihoods can be scaled by an appropriate parameter π_{FP} , π_{link} , π_{init} or π_{term} . The setup of these parameters allows a direct manipulation of the linking hypothesis. For example, increasing π_{init} or π_{term} relative to π_{link} makes the system more conservative in linking tracklets. The added benefit of these scaling parameters is that they allow the reuse of the trained classifier for somewhat different datasets, without the need to annotate them and retrain the classifier.

4.7 Features for the linking classifier OUTLINE

In this section we present an overview of the visual and spatio-temporal features implemented and tested for the classifier for linking tracklets. In section 4.7.3 we enumerate which of these features have been selected to be used in the final classifier.

Cell feature descriptor The difference of vectors containing appearance features obtained from the cell detection module for each candidate linking tracklet. A description of these features is available in section 3.5.

Gap size The number of frames between the tail and head of a pair of tracklets.

Position distance A two dimensional vector containing the absolute distance between the head and tail of a pair of tracklets corresponding to the x and y coordinates.

Square of position distance Same as *Position distance*, but the value is squared.

Distance between points The euclidean distance between the positions of the head and tail of linking tracklets. This is similar to the previous features, but combines the distance between both coordinates into a single value.

Direction angle The difference between the tracklet orientations computed from the last few frames of the first tracklet and the first few frames on the second tracklet. The number of frames used to compute the direction is parametrized.

Orientation variance Similar to the previous feature, but computes the difference of variance of orientation change within tracklets.

Mean displacement The difference of mean displacement between cell detections within the tracklets.

Displacement variance Similar to the previous feature, but computes the difference of variances of the displacement between cell detections.

Distance from edge The difference between minimal and maximal distances of the head or tail of the tracklets from the edge of the field of view.

Gaussian broadening distribution This feature is detailed in section 4.7.2.

Augment this list as more features are added

4.7.1 Estimating the velocity with Kalman filters NEW

This feature has not yet been implemented

4.7.2 Gaussian broadening feature DRAFT I

This section describes a feature that is based on the observation of the motion of the trajectories. It makes intuitive sense that the probability of linking two tracklets is inversely proportional to the distance (number of frames) between tracklets. Additionally we can observe that a cell will not travel in a perfectly straight line, but might deviate from it. Using these observations, we have devised a feature inspired by Doppler broadening ¹.

After a tracklet ends, its motion is extrapolated for several frames. The probability of a tracklet being at each point along the extrapolated tracklet is assumed to be normally distributed. Because of the observation that a tracklet might deviate from its temporary direction, the variance is assumed to be larger at each next extrapolated location. If we place such normal distributions along the extrapolated trajectory, and we normalize their value such that the sum of all the normal distributions is equal to one, we obtain a new distribution that describes the probability of a tracklet's position in the future.

We can use this tracklet's distribution to evaluate the likelihood that another candidate tracklet should be linked to it by summing the values of the distribution at the locations corresponding to the cell detection of the candidate tracklet.

Figure 4.5 shows an example profile of such a distribution, together with two candidate linking tracklets. The sum of values from the distribution at the location of the tracklet's cell detections indicates that the black tracklet is more likely to be linked to the red one than the green tracklet. The figure also shows that the distribution correctly adapts to tracklets composed of a single cell detection, or similarly to tracklets with very little movement.

4.7.3 Best feature selection NEW

This feature has not yet been implemented

¹http://en.wikipedia.org/wiki/Doppler_broadening

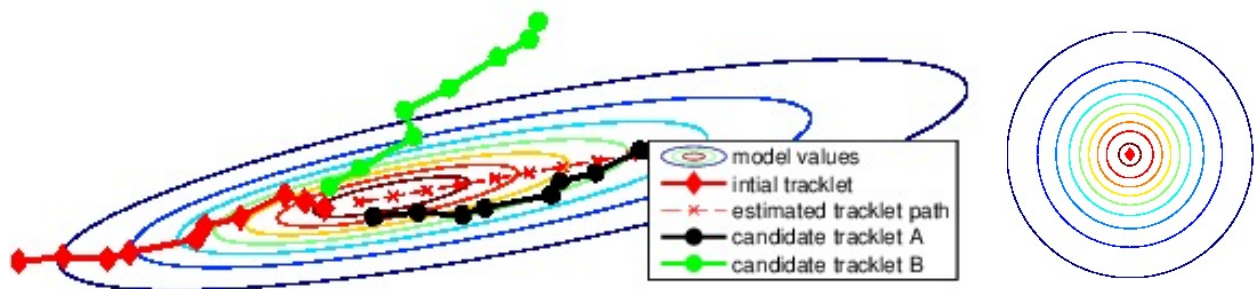


Figure 4.5: Gaussian broadening feature. The distribution is computed for the red tracklet. The contour colours of the distribution are red for higher values and blue as the value decreases. The candidate black tracklet is given a linking likelihood of 0.20, while the green, which is angled from the direction of the red tracklet, a value of 0.06. On the plot on the right we see that if a tracklet is composed of only one cell detection the value of the feature is equal all around it.

Define how the best features have been selected

Describe which features performed best

4.8 Implementation details NEW

This feature has not yet been implemented

Answer: what was the problem, how we solved it. what are the consequences

- we discard $\text{plink} < 0.5$ - we trim number of hypothesis: how - we can perform the method iteratively: reduce memory usage, number of hypothesis. -

Appendices

Bibliography

- [1] P. K. Elzbieta Kolaczowska, “Neutrophil recruitment and function in health and inflammation,” 2013. 8
- [2] J. Pillay, I. den Braber, N. Vrisekoop, L. M. Kwast, R. J. de Boer, J. A. M. Borghans, K. Tesselaar, and L. Koenderman, “In vivo labeling with 2h2o reveals a human neutrophil lifespan of 5.4 days,” *Blood*, vol. 116, no. 4, pp. 625–627, 2010. 8
- [3] P. S. Tofts, T. Chevassut, M. Cutajar, N. G. Dowell, and A. M. Peters, “Doubts concerning the recently reported human neutrophil lifespan of 5.4 days,” *Blood*, vol. 117, no. 22, pp. 6050–6052, 2011. 8
- [4] C. Arteta, V. Lempitsky, J. A. Noble, and A. Zisserman, “Learning to detect cells using non-extremal regions,” in *Proceedings of the 15th International Conference on Medical Image Computing and Computer-Assisted Intervention - Volume Part I*, MICCAI’12, (Berlin, Heidelberg), pp. 348–356, Springer-Verlag, 2012. 9, 10, 12, 17, 18, 19, 20, 22, 48, 53, 62
- [5] Y. Chen, K. Biddell, A. Sun, P. Relue, and J. Johnson, “An automatic cell counting method for optical images,” in *[Engineering in Medicine and Biology, 1999. 21st Annual Conference and the 1999 Annual Fall Meeting of the Biomedical Engineering Society] BMES/EMBS Conference, 1999. Proceedings of the First Joint*, vol. 2, pp. 819 vol.2–, Oct 1999. 11
- [6] X. Chen, X. Zhou, and S.-C. Wong, “Automated segmentation, classification, and tracking of cancer cell nuclei in time-lapse microscopy,” *Biomedical Engineering, IEEE Transactions on*, vol. 53, pp. 762–766, April 2006. 11, 14
- [7] L. Vincent, “Morphological grayscale reconstruction in image analysis: applications and efficient algorithms,” *Image Processing, IEEE Transactions on*, vol. 2, pp. 176–201, Apr 1993. 11
- [8] J. Serra, *Image Analysis and Mathematical Morphology*. Orlando, FL, USA: Academic Press, Inc., 1983. 11
- [9] D. Mukherjee, N. Ray, and S. Acton, “Level set analysis for leukocyte detection and tracking,” *Image Processing, IEEE Transactions on*, vol. 13, pp. 562–572, April 2004. 12, 13
- [10] C. Tang, Y. Wang, and Y. Cui, “Tracking of active cells based on kalman filter in time lapse of image sequences of neuron stem cells.” 12, 15
- [11] D. Xu and L. Ma., “Segmentation of image sequences of neuron stem cells based on level-set

- algorithm combined with local gray threshold.,” Master’s thesis, Harbin Engineering University, 2010. 12
- [12] C. Arteta, V. S. Lempitsky, J. A. Noble, and A. Zisserman, “Learning to detect partially overlapping instances.,” in *CVPR*, pp. 3230–3237, IEEE, 2013. 12, 13, 19
 - [13] J. Matas, O. Chum, M. Urban, and T. Pajdla, “Robust wide baseline stereo from maximally stable extremal regions,” in *Proceedings of the British Machine Vision Conference*, pp. 36.1–36.10, BMVA Press, 2002. doi:10.5244/C.16.36. 12
 - [14] T. Joachims, T. Finley, and C.-N. J. Yu, “Cutting-plane training of structural svms,” *Mach. Learn.*, vol. 77, pp. 27–59, Oct. 2009. 12
 - [15] R. Bise, T. Kanade, Z. Yin, and S. il Huh, “Automatic cell tracking applied to analysis of cell migration in wound healing assay,” in *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*, pp. 6174–6179, Aug 2011. 24
 - [16] S. Huh, *Toward an Automated System for the Analysis of Cell Behavior: Cellular Event Detection and Cell Tracking in Time-lapse Live Cell Microscopy*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, March 2013. 14, 55
 - [17] D. House, M. Walker, Z. Wu, J. Wong, and M. Betke, “Tracking of cell populations to understand their spatio-temporal behavior in response to physical stimuli,” in *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*, pp. 186–193, June 2009. 14
 - [18] B. Xu, M. Lu, P. Zhu, Q. Chen, and X. Wang, “Multiple cell tracking using ant estimator,” in *Control, Automation and Information Sciences (ICCAIS), 2012 International Conference on*, pp. 13–17, Nov 2012. 14
 - [19] K. Li and T. Kanade, “Cell population tracking and lineage construction using multiple-model dynamics filters and spatiotemporal optimization,” in *Proceedings of the 2nd International Workshop on Microscopic Image Analysis with Applications in Biology (MIAAB)*, September 2007. 15
 - [20] A. Massoudi, D. Semenovich, and A. Sowmya, “Cell tracking and mitosis detection using splitting flow networks in phase-contrast imaging,” in *Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE*, pp. 5310–5313, Aug 2012. 15
 - [21] L. Zhang, Y. Li, and R. Nevatia, “Global data association for multi-object tracking using network flows,” in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pp. 1–8, June 2008. 15, 27
 - [22] C. Huang, B. Wu, and R. Nevatia, “Robust object tracking by hierarchical association of detection responses,” in *Computer Vision - ECCV 2008* (D. Forsyth, P. Torr, and A. Zisserman,

- eds.), vol. 5303 of *Lecture Notes in Computer Science*, pp. 788–801, Springer Berlin Heidelberg, 2008. 16, 27
- [23] R. Bise, Z. Yin, and T. Kanade, “Reliable cell tracking by global data association.,” in *ISBI*, pp. 1004–1010, IEEE, 2011. 16, 24, 27, 30, 55, 62
- [24] H. Kuhn, “The hungarian method for the assignment problem,” *Naval Research Logistics Quarterly*, vol. 2, pp. 83–97, 1955. 16
- [25] J. Matas, O. Chum, M. Urban, and T. Pajdla, “Robust wide-baseline stereo from maximally stable extremal regions,” *Image and Vision Computing*, vol. 22, no. 10, pp. 761 – 767, 2004. British Machine Vision Computing 2002. 18
- [26] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun, “Support vector machine learning for interdependent and structured output spaces,” in *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML ’04, (New York, NY, USA), pp. 104–, ACM, 2004. 20
- [27] K. Li, E. D. Miller, M. Chen, T. Kanade, L. E. Weiss, and P. G. Campbell, “Cell population tracking and lineage construction with spatiotemporal context,” *Medical Image Analysis*, vol. 12, no. 5, pp. 546 – 566, 2008. Special issue on the 10th international conference on medical imaging and computer assisted intervention - {MICCAI} 2007. 24, 64
- [28] M. Looney, E. Thornton, D. Sen, W. Lamm, R. Glenney, and M. Krummel, “Stabilized imaging of immune surveillance in the mouse lung,” *Nature Methods*, vol. 8, no. 5, pp. 91–6, 2011-01-01 00:00:00.0. 39
- [29] S. Blackman and R. Popoli, *Design and Analysis of Modern Tracking Systems*. Artech House radar library, Artech House, 1999. 55
- [30] S.-i. S. Eom, K. D. F. Elmer, B. Ryoma, and K. Takeo, “Tracking of hematopoietic stem cells in microscopy images for lineage determination,” *J Latex Class Files*, vol. 6, no. 01-09, p. 9, 2007. 55