Development of detection and tracking methods for characterizing Cardiac Progenitor Cell migration patterns in *Gallus gallus*

Connor Reynolds

Student Number: 100127017

Primary supervisor: Simon Moxon

Second supervisor: Andrea Munsterberg

Other supervisors: Leighton Folkes and Timothy Grocott

School of Biological Sciences, University of East Anglia

Word count: 23193

Acknowledgements

I would firstly like to thank my leading supervisors Simon Moxon and Leighton Folkes for their advice and mentoring throughout my MScRes. I would also like to thank my additional supervisors Andrea Munsterberg and Timothy Grocott for their useful input to the project. I would finally like to thank my partner Huang Su for her support during my write-up.

Contents

Abstract

Cardiac cells originate from a population of progenitors which establish the early heart tube following a long-range migration event that initiates during gastrulation. After exiting the mid-anterior primitive streak, the trajectories of cardiac progenitor cells (CPCs) up to the fusion of the primitive heart tube have been previously characterised. Associating the function of signalling molecules with these migration patterns and the subsequent outcome on heart development is still poorly understood. The popular and versatile software TrackMate has previously been used to track fluorescently labelled cells in embryos across time-lapsed images, but the tool has not been evaluated in the context of CPC migration in chick embryos. Here, we identified shortcomings in TrackMate's cell detection and single-cell-tracking methods in general, so instead developed a tracking method based on kernel density estimates of generalized groups of cells. After training and evaluating our approach using simulated migration datasets, we applied the method on true CPC time-lapses to determine a significantly more obtuse initial exit angle in CPCs treated with follistatin compared to controls. We further provided a proof-of-concept machine learning-based workflow for automatically comparing CPC migration time-lapses when current datasets are expanded.

Chapter 1. Background to cardiac progenitor cells

1.1 Introduction

The development of the heart as the first functional organ is shared amongst all amniote embryos and is initially formed during early embryogenesis as a linear structure called the heart tube, which progressively develops into a more mature heart through remodelling of the tube [1]. The cardiomyocytes that form the heart tube originate from the mesoderm-fated cardiac progenitor cells (CPCs). During gastrulation, CPCs ingress through the primitive streak before migrating bilaterally towards the medial-anterior of the embryo, where they contribute exclusively to the splanchnic layer of the lateral plate mesoderm as bilateral heart fields [2]. During neurulation, a cardiogenic crescent establishes, where both heart fields migrate medially towards one another while the CPCs differentiate into endocardial tubes, which subsequently fuse to form a single heart tube [3, 4].

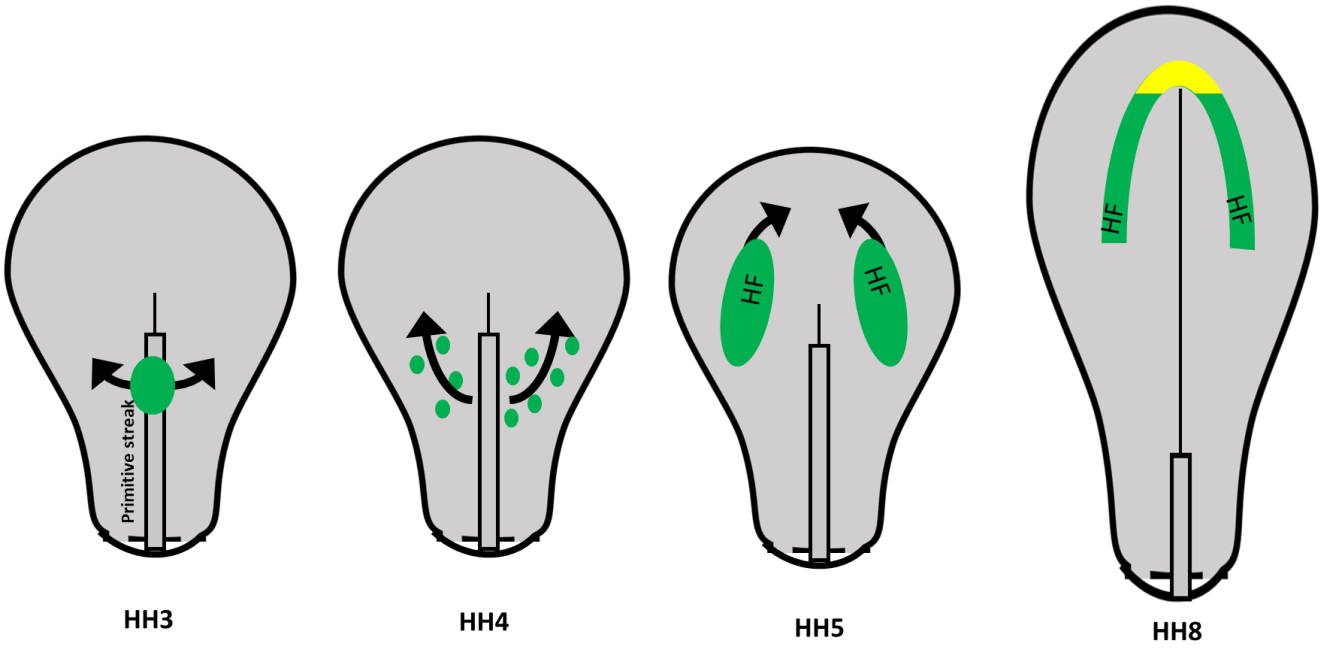


Figure 1.1 Migration of CPCs (green) at different Hamburg-Hamilton (HH) (1951) stages. At HH3 CPCs emerge from the mid-anterior primitive streak and migrate bilaterally towards the medial-anterior of the embryo. At HH4 the CPCs cease emerging from the primitive streak. At HH5, the cells establish bilateral heart fields specifically in the splanchnic layer of the lateral plate mesoderm, which to move medially. At HH8 the heart fields form a cardiac crescent, though it is disputed as to whether this is a single crescent or if it is separated (yellow).

The regulated early development of the heart is a critical step in embryogenesis, as congenital defects in heart tube formation can lead to cardiac dysfunction and often prenatal death of the organism such as cardia bifida, where the heart tube fails to fuse. As CPCs are the main contributors to the heart tube, their behaviour must be tightly controlled throughout early embryogenesis by morphogens or other signalling molecules. Numerous studies have shown the upregulation or knockout of different signals can result in defective heart tube fusion, with some demonstrating this is paired with abnormal CPC trajectories [5, 6]. As the complex network of converging signals that regulate CPC movement is poorly understood, computational techniques used to analyse and quantify their migratory behaviours as a phenotype across different signal treatments must be regularly benchmarked and improved when necessary.

1.1.1 The chick as a model organism for cardiac development

Chicken (*Gallus gallus*) embryos are a popular experimental model organism for investigations in developmental biology, with their use predating modern molecular techniques. Early observations in developing chick embryos identified the heart as the first functional organ, defined embryos into their respective germ layers – the endoderm, mesoderm and ectoderm, and were used to support the prevailing theory that organisms develop progressively from a simpler embryonic state as opposed to being small, preformed adults (reviewed by Stern, 2005). Their continued use since the onset of modern experimental embryology has led to numerous landmark discoveries in amniote development including the characterisation of migratory patterns of epiblast derived cells in gastrulation [7, 8], the induction of the primitive streak by the hypoblast and the identification of the Hensen's node as the organizer region in amniotes [9].

Chick embryos are an attractive model for studying early cardiac development for numerous reasons, firstly being they develop *in ovo*, which makes them more accessible to manipulations compared to organisms that develop *in utero*. This means early chick embryos can be readily separated from the egg and allowed to develop in culture [10], making them even more exposed to interventions as well as allowing for their analysis through the use of microscopy [5, 6, 11]. Additionally, like other nonrodent amniotes, the embryonic precursor cells of the chick blastocyst form as a flat and disk-shaped epiblast that allows for investigations to clearly observe cell movement and fate during gastrulation [12].

The entire development of a normal chick embryo from egg laying to hatching was categorised into a series of 46 stages by *Hamburger and Hamilton* in 1951 [13], and more recently a review by *Martinsen* [14] mapped developmental landmarks in heart formation to each Hamburger-Hamilton (HH) stage, which is beneficial for researchers to use as a reference. The early stages of cardiac development are also highly conserved between chicks and humans, both of which mature into a four-chambered heart [15], meaning insights derived from the former can be highly applicable to understanding heart development and defects in humans.

1.1.2 Regulation of CPC migration and fate

The fate of avian CPCs is established pre-gastrulation, as cardiogenic precursors have been mapped to the epiblast prior to primitive streak induction [16], where they are specified by activin-like signals from the hypoblast and become mostly committed to a cardiogenic fate by HH3 [17]. Like other mesodermal cells, the migration of CPCs away from the mid-anterior primitive streak at HH4 is regulated by a chemo-attraction and -repulsion to FGF4 expressed at the anterior and FGF8 at the posterior respectively [18]. This chemotaxis towards the lateral plate mesoderm is supported by Wnt3a expression at the primitive streak and Wnt8c across the posterior region of the embryo, which both inhibit further specification of CPCs [19]. Wnt3a also guides their movement through chemo-repulsion [5].

Following the establishment of the heart fields at HH5, the formation of the epicardial tubes and their subsequent fusion at HH10 requires the promotion of CPC differentiation by signals at the anterior endoderm, including the Wnt inhibitor crescent [19] and the bone morphogenetic proteins (BMP) BMP2 and BMP4, which are known to induce the cardiogenic transcription factor Nkx2.5 [20]. The dysregulation of these signals can disrupt cardiogenesis, with overexpression of Wnt3a near the lateral plate mesoderm or BMP2/4 near the mid-primitive streak leading to significantly wider CPC migration trajectories that results in cardia bifida [5, 6]. Additionally, if BMP2/4 signals are not inhibited by chordin at the mid-anterior primitive streak prior to gastrulation, cardiogenesis does not occur [21].

Other signals such as the activin antagonist follistatin can inhibit the specification of CPCs prior to gastrulation [17, 21], but it is not known what effect it has during their migration after HH3. Overexpression of retinoic acid also interferes with the establishment of the cardiogenic crescent and can result in cardia bifida, possibly due to the fact it activates non-canonical Wnt pathways such as Wnt3a [22, 23].

Understanding how different signal pathways control CPC migration using gain- or loss-of-function interventions in chick embryos could offer important insights in the causes of cardia bifida or non-fatal congenital heart defects in humans. Additionally, as they are candidates for cardiac cell therapies, knowledge of regulating CPC behaviours may help in the development of new techniques to treat heart failure [10].

1.1.3 Analysis of CPC migration

Localising CPCs is critical when studying their migration, as in standard brightfield microscopy images they are indistinguishable from the rest of the developing chick embryo. Labelling of the cardiogenic mesoderm with iron oxide was used in early investigations to show the establishment of the bilateral heart fields and their coinciding fusion with heart tube formation [24], following which radiolabelling was used to develop a refined cardiogenic fate map from HH5 to the fusion of the heart tube at HH10 [3]. Later studies using antibody labelling of grafted chick-quail chimeric cells showed CPCs leave the mid-anterior primitive streak at HH3 [2], where they migrate bilaterally towards the lateral plate mesoderm, which has more recently been demonstrated in real-time from fluorescent labelling of CPCs at the mid-anterior primitive streak with fluorescent proteins [5, 6].

Time-lapsed-microscopy (TLM) involves taking a series of images at specific time-points, which can be generated using microscopy techniques such as fluorescent microscopy imaging (FMI), where light of only specific wavelengths are detected [25], phase contrast (PC) imaging and differential interference contrast (DIC) imaging, both of which use the interference of light travelling through objects to generate images

[26]. TLM-imaging of whole chick embryos has been used since the general "polonaise" movement of cells during gastrulation was first characterised [7, 8], which continued throughout the development of the aforementioned cardiogenic fate maps with early isotopic labelling [3], but these maps were only able to investigate the fate and movement of CPCs after they have established in the splanchnic layer of the lateral plate mesoderm at HH5.

Recent advancements in cell localisation have allowed for specific embryonic cell lineages to be labelled during gastrulation using electroporation to transfect GFP expression plasmids into HH3 primitive streak cells of a donor chick embryo, after which the GFP-labelled cells are transplanted into different primitive streak regions of a host embryo, where their movement and fate can be observed [18]. Through this method, it is now possible to generate fluorescent TLM-images that can analyse GFP-labelled CPC trajectories across their entire migration from their departure of the primitive streak at HH3 to the fusion of the heart tube at HH10, which was used previously to determine the effects of Wnt3a and BMP2/4 expression in CPC movement [5, 6]. Recent techniques have been developed to fluorescently label primitive streak cells by transfecting expression plasmids directly to study somitogenesis [27], avoiding the need for invasive cell transplantations.

Tracking CPCs is often performed using advanced techniques across the TLM-movie, where each greyscale image is represented as a matrix containing intensity values from 0-255. While it is necessary for an image matrix to contain at least two spatial dimensions (2D) matrices can be arrayed to produce images with further dimensions. Many investigations have quantified morphological features of developing embryos with the addition of another spatial plane in three dimensions (3D), but 3D imaging may also refer to an array of matrices taken across different time points, as in TLM-movies, and time-lapsed arrays of spatially 3D images are often considered four-dimensional (4D) [28]. To avoid confusion in this terminology, the dimensionality of a single or time-lapsed-series of images will be referred to according to only their spatial dimensions, in which our TLM-images is 2D.

1.2 Cell Tracking overview

Tracking objects-of-interest across a time-lapsed image-series involves their detection and linkage between image frames, which are usually independent algorithms performed sequentially. For cell tracking, the approaches used by these algorithms can vary significantly depending on the resolution of the images, the microscopy techniques used and the environment of the cells during the imaging.

The first step of a cell tracking method is to detect the cells by distinguishing them as objects-of-interest, referred to as the foreground, from regions of no interest – the background. In TLM-movies, this is performed for each image frame and, if successful, allows spatial features of each cell to be extracted for the linkage step. This step can be particularly challenging when the lighting in images is uneven or if they are saturated in noise, as this can lead to regions of the background being falsely detected as cells.

Linking cells across image-frames automatically can be computationally costly, algorithmically non-trivial and errors are rarely able to be avoided entirely, meaning that in TLM-movies with a small number of cells across a few frames, manual tracking is often preferred [29, 30]. In an image-series with hundreds of cells across hundreds of frames however, manual tracking would be time consuming to the point a method that is at least partially automated is required. Algorithms designed for automatic cell linkage are generally placed into two categories: model evolution, where the boundaries of detected cells are defined then linked to cells in other frames based on their degree of overlap, and detection association, as shown in figure 2, where cells are linked to others across the image-series based on their relatedness. The latter is more common and almost universally involves distal proximity of cells, but they can include closeness of other features such as size or shape.

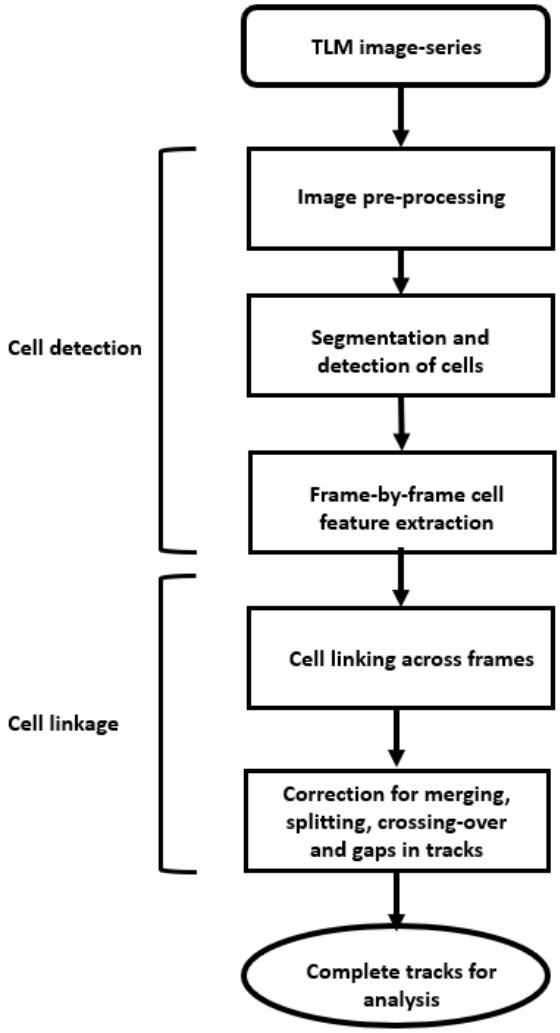


Figure 2. General approach for detection association-based cell tracking methods, involving algorithms for detecting cells in each frame following by cell-linkage. Adapted from *Hamilton, 2009.*

The increased use of TLM techniques in biomolecular research to quantify cellular, subcellular or extracellular movement patterns means there is a growing demand for improved tracking algorithms which has led to a series of competitions by the IEEE International Symposium on Biomedical Imaging – the Cell Tracking Challenges (CTCs) and Particle Tracking Challenges (PTCs), where participating researchers attempted to develop high-performing tracking pipelines for cells or particle-like signals in TLM-movies produced using different microscopy techniques across several biological contexts. The most competitive tracking pipelines produced for the CTCs and PTCs have been reviewed by *Ulman et al.* [31] and *Chenouard et al.* [32] respectively, both of whom showed no specific approaches were versatile enough to perform optimally across all types of images and biological processes.

It is important to note that algorithms referred nominally by the challenges as "cell trackers" are often considered distinct from "particle trackers", which are designed for tracking intracellular particle-like signals and do not consider mitotic events. In some TLM-movies of fluorescently labelled cells however, the approaches used by "particle trackers" may be superior for cell detection and linkage than nominal "cell trackers". This is because poorly resolved cells often appear particle-like as opposed to the more detailed cells in highly resolved images, which often have clearly visible spatial features and intracellular structures that can be used in the linkage step.

CTC algorithms based on detection association have similar frameworks with those in the PTC, with many of the approaches used for detection and linkage being identical, the major deviance of which being most cell trackers detected division while particle trackers did not. The similarities between "cell" and "particle trackers" are further demonstrated by the fact TrackMate, a popular and widely used tracking tool that is often considered a "particle tracker", can optionally detect division events and has been used for both single-cell [28] and intracellular molecule tracking [33]. For this reason, we shall consider any method which can be used for cell detection and linkage as a cell tracker.

While there is an abundance of cell tracking algorithms, most are available only as low-level libraries or source code and using them to their full potential could require extensive computational experience which many researchers do not possess. As the development of evermore competitively performing algorithms has grown, so has the demand for them to be built into software tools designed so researchers with little-to-no programming or mathematical experience can use them. A commentary by *Carpenter et al.* [34] set several criteria for bioimaging tool usability, recommending they should aim to be user-friendly, publicly available as a functional, tested and benchmarked software, modular for specific purposes, interoperable with other imaging or analysis software and have an accessible open-source code that is written idiomatically.

1.2.1 Existing cell tracking tools

The availability of a cell tracking tool is possibly the most important of the *Carpenter et al.* criteria [34], as limiting the access to a tool means it is less likely to be used, documented and benchmarked compared to other more accessible tools. As shown in Table 1, many cell trackers, such as ImarisTrack [35] and Volocity (Quorum Technologies) are restricted behind a paywall, the expensive access of which may discourage their use by researchers. Many freely distributed tracking software used in previous studies also fail this criterion, with tools such as CellTrack [36], CellCognition [37], TimeLapseAnalyser [38] and TLM-tracker [39] being either unsupported for modern operating systems or inaccessible for public use. Another failed criterion in all paid and some free tracking tools is the lack of open-source code for the software, which may be advantageous for researchers to review or improve the performance and power of the tracking algorithm. Some software are fully available, but appear to not function as intended, such as NucliTrack [40].

Notable cell tracking tools that meet much of the usability criteria by being freely available, modular, open source, interoperable and functional in modern operating systems include the TrackMate [41] plugin for the software ImageJ/Fiji, CellProfiler with the TrackObjects module [42] and Spot Tracker, a plugin for the software Icy [43]. These tools can be considered semi-automated as they require user input for optimal cell detection and tracking, which in high-throughput or batch processing could be problematic as TLM-image stacks can vary in quality and thus require different inputs for optimisation.

Table 1.1 Accessibility and functionality of different software tools for cell tracking. Green indicates potential usability, red indicates non-usability

Software	Accessibility				Supported OS		Functionality	
	Free	Publicly available	Open source	Installs compiler-free	Windows 10	Ubuntu 18.04	Runs without crashing	Language
Fiji (ImageJ) + Trackmate	✓	✓	✓	✓	✓	✓	✓	Java
CellProfiler	✓	✓	✓	✓	✓	✗	✓	Python
Icy - Spot Tracker	✓	✓	✓	✓	✓	✓	✓	Java
Nuclitrack	✓	✓	✓	✓	✓	✗	✗	Python
CellCognition	✓	✓	✓	✓	✗	✗	?	Python
CellTrack	✓	✓	✓	✓	✗	✗	?	C+
TimeLapseAnalyzer	✓	✓	✓	✓	✗	✗	?	MATLAB
CellTracker	✓	✓	✓	✗	✓	✗	?	MATLAB
LEVER	✓	✓	✓	✗	✓	✗	?	MATLAB
TLM-tracker	✓	✗	✗	✗	✗	✗	?	MATLAB
ImarisTrack	✗	✓	✗	?	✓	✗	?	?
Volocity	✗	✓	✗	?	✓	✓	?	?

TrackMate is a tracking tool that uses particle tracking-based principles within its algorithm, but it has the option for lineage tracing and has been used previously to track singe-cells [28, 41]. Having over 1000 citations across various studies, TrackMate's overall method is considered versatile and robust, using a

blob detection approach followed by a frame-to-frame particle-linking algorithm called the linear assignment problem (LAP) solution, which is based on a mathematical formulation by *Jaqaman et al.* [44]. Being a plugin for ImageJ/Fiji, there are numerous pre- and post-processing tools that can be used to complement or correct the cell tracking and, as Bioformats is integrated into ImageJ, TrackMate can also analyse imaging data of many file types without formatting issues, including videos, image stacks and single image files.

The detection stage requires the user to define an estimated radius of the cells, which can be problematic if cell sizes are different. Also, the performance of the cell-linkage phase can only be fully optimised by manually setting parameters such as the maximum linkage distance, but users may not be able to determine which values will generate the most accurate tracks. While the semi-automated design of TrackMate could be argued to be beneficial to allow fine-tuning, it could be considered less user-friendly than a fully automated method. Requiring manual input for each image is also problematic when analysing batch or high-throughput imaging data, as each TLM-movie may require different parameters.

CellProfiler is a popular bioimaging software proclaimed to be designed specifically for high-throughput image analysis while also possessing a user-friendly interface, containing multiple guided pipelines for quantifying cell features in images such as cell tracking in TLM-movies [42]. Unlike TrackMate, CellProfiler takes only image stacks and cannot analyse TLM-image files in video format directly, of which the conversion of the latter into the former is time consuming and storage intensive. The cell detection algorithm involves binarization of the image and labelling of each cell, requiring input from the user to select the segmentation method. Like TrackMate, the LAP algorithm [44] is used for cell-linkage and the parameters must be set by the user. This suggests CellProfiler may suffer from the same disadvantages in usability and analysing images in batches as TrackMate.

SpotTracker is a particle-based tracking plugin for Icy [43], a bioimaging software similar to ImageJ/Fiji that offers the same versatility in analysing imaging files across different formats. The tracking algorithms requires more user input than TrackMate and CellProfiler which use a wavlet transform to detect spots only of user-defined sizes before the multiframe cell-linkage multiple hypothesis tracking (MHT) algorithm is used, as described by *Chenouard et al.* [45]. SpotTracker appears to be designed with user-control in mind, as it offers options allowing users to manually define which cells are to be tracked.

While the aforementioned software use algorithms that are designed to be versatile, the Tracking Challenges demonstrated no single cell tracking method can perform optimally across all types of imaging data [31, 32]. The software also only generates tracks, usually as labelled cartesian coordinates, and do not provide methods for analysing them quantitatively, while supplemental software developed to analyse cell track data often focus on cell lineages, rather than their movement [46]. For these reasons, an approach designed specifically for tracking fluorescently labelled cells in whole-embryo fluorescent TLM-images of nonrodent amniotes could outperform present software while also having an incorporated algorithm to analyse the data, thus allowing for more a more quantitative analysis of CPC migration.

1.3 Project aims

The objectives of this project were to assess the performance of existing cell tracking software in capturing the migration of GFP-labelled CPCs in fluorescent TLM-movies of the early chick embryo and to identify potential shortcomings in both cell detection and linkage when generating cell trajectories. We aimed to overcome these flaws by testing a variety of computer vision and machine learning techniques, from which we will select the highest performing approaches for our specific TLM-movies, which will then be used to quantitively characterise differences in CPC migration patterns in chick embryos with varying gain- or loss-of-function interventions. Finally, we developed our algorithms into a frame-work designed to meet many of the automated criteria defined by *Carpenter et al.* [34] for bioimaging software.

All TLM-movies used in this project were provided by the Münsterberg group, members of which also contributed to their generation as well as the culturing and labelling of chick embryos, as described in Chapter 2. The limitations of existing cell tracking software and the need for tools designed specifically for tracking fluorescent cells across TLM-images of whole embryos was first noted by the group, which coincides with the conclusions by *Chenouard et al.* [32] and *Ulman et al.* [31] that no single tracking algorithm has been shown to perform optimally across all imaging data and biological contexts.

Python 3.7 [47] was chosen to design the tool due to it being a widely used programing language for computer vision and data analysis while possessing an intuitive syntax for more idiomatic code. It also supported by many powerful libraries such as OpenCV for general computer vision techniques, Numpy for array and matrix operations [48], and scikit-learn [49] for machine learning algorithms, which are all compiled modules that can perform efficiently. OpenCV offers basic functions for image processing [50], including blurring for noise filtering, global and adaptive techniques for thresholding, and morphological operations for watersheds.

The results of my thesis are presented across different chapters with each focusing on a different stage of the research project. More detailed descriptions of some methods are described in a supplementary Github: https://github.com/ConnorReynoldsUK/MScRes_thesis. Chapter 2 demonstrates the challenges when using common cell tracking tools and suggests they are not suitable for our biological context. We therefore set out to implement our own cell tracking method.

Chapter 3 explores using different methods for cell detection, as accurate positional information is required for accurate tracking. Our custom convolution filter for segmentation and cell centroid extraction can be found accessed via the Github notebook **Chapter3_SEGMENTATION.ipynb**.

Chapter 4 introduces the approach of tracking cells as generalized groups using probability density estimates. To ensure our methods can be evaluated, we generated simulated movies with an established ground-truth trajectory as shown in the notebook **Chapter4_simulation.ipynb**. We then implemented two tracking methods, Frametrack (**Chapter4_Frametrack.ipynb**) and Mudtrack (**Chapter4_Mudtrack.ipynb**).

In Chapter 5, we show how Mudtrack generated trajectories can be used to extract a phenotype in the form of exit angle to compare between experimental groups. We also explore other methods for comparing these groups using Trackmate and machine learning models.

Chapter 2. Evaluating TrackMate

2.1 TrackMate overview

Of the numerous algorithms for cell detection and tracking described in Chapter 2, TrackMate is potentially the most widely used tool for the spatiotemporal analysis of TLM-images, possessing over 1000 citations in literature involving cell or particle tracking across various biological contexts [41]. It is a default plugin for Fiji/ImageJ [51], a popular software which provides multiple out-of-the-box tools in image processing and analysis for users at all levels and is designed to be user-friendly so researchers with little computational experience can use it. The cell detection and tracking methods were developed to be robust in that they are suitable for multiple types of microscopy images and cell types in datasets with varying degrees of quality.

Using approaches intended to identify objects as “spots” (or intensity signals) irrespective of their shape and linking them across all frames, TrackMate is considered a single-particle tracker, though the tool has been previously used for spatiotemporal analyses of both subcellar molecules and cells in both 2D and 3D space [28, 52–54]. As the algorithms for spot detections and linkages are dependant on user-defined parameters, the software can be considered semi-automated, with the detection stage requiring defined values for a quality threshold and an object radius, while the tracking stage requires manual input for maximum linkage distances with the allowance of gap closing and object splitting or merging.

TrackMate’s Spot Detector includes several convolution-based methods for object detection, but the most commonly used is the Laplacian of Gaussian (LoG) detector, where a convolutional filter is first applied using a sigma with a kernel size proportional to the radius provided, following which points of local maxima are detected and assigned a quality value based on the intensity and the closeness between the object diameter and the provided diameter. When two detections have a distance closer to that of the provided radius, the one with the lower quality is discarded. Using a quadratic function of the local pixel intensities surrounding the initial detection, which was inspired from a study by *Lowe* [55], a more accurate position of the cell and quality value is predicted by finding the peak of the parabola, which is localized to a sub-pixel resolution, following which detections with quality values lower than the user-defined threshold are discarded.

TrackMate uses a spot linkage approach inspired by the linear assignment problem (LAP) method developed by *Jaqaman et al.* [44], in which multiple costs are calculated to determine if cells are to be linked across a frame, if a gap closing event is to occur or when cells split or merge. The ‘problem’ of the LAP method refers to the computational issue in assigning optimal costs for each event, which can be highly costly using a brute-force approach when there are multiple possible assignments, thus TrackMate solves the LAP mathematically using a greedy algorithm developed by *Munkres* [56] to find the assignment series with the smallest sum of costs.

Despite having been used extensively, TrackMate has limited use in tracking cardiac progenitor cells (CPCs) in chick embryos. The landmark spatiotemporal investigation on CPC migration by *Song et al.* [6], alongside preceding investigations in the movement of primitive streak derived cells developing *Gallus gallus* embryos [5, 18], involved tracking based on overlaying sequential frames to generate a trajectory rather than a single-particle-tracking based approach. A more recent study on cell movement in chick embryo somite morphogenesis did use TrackMate, but the images analysed were spatially 3D, contained relatively well resolved cells that had distinguishable boundaries, and pre-processing was used prior to running the tool [28].

We initiated our investigation in characterising migrating CPC trajectories in TLM-images of fluorescently labelled chick embryos by first testing the suitability of using TrackMate for the analysis, where we wanted to determine how accurate the tool was for cell capture and tracking across frames, how consistent the optimal user-defined parameters were across different movies, and if there were significant differences between movies expressing different fluorescent markers. We further sought to use the results from this analysis to suggest solutions to potential limitations in TrackMate's detection or linkage methods and to establish how suitable single-particle-tracking methods are in general for our TLM-movies.

2.2 Materials & methods

2.2.1 Embryo culturing and GFP-labelling

Fertile brown chicken eggs were grown in a humidified incubator at 38^{o}C until reaching HH3, following which the embryos were harvested and cultured as described by *Chapman et al.* [10]. The mid-anterior primitive streak where CPCs are localised was microinjected with IRES-GFP expression plasmids following protocols described by *Yue et al.* [5]. Expression plasmids were transfected into CPCs through electroporation using methods described by *Yang et al.* [18], following which embryos were left to heal overnight.

2.2.2 Time-lapsed-microscopy of embryos

GFP-labelled embryos were cultured in six-well plates, after which long-term time-lapsed-image-series were generated using an inverted wide-field microscope (Axiovert, Ziess), with brightfield and fluorescent images being captured at 5-minute time-points across 24 hours. Time-lapses from HH3-4 up to HH8-10 were exported as ZVI files containing images that were exclusively greyscale.

2.2.3 Software for cell detection

Raw ZVI TLM-movies were imported into ImageJ with the Fiji plugins, following which the brightfield and fluorescent channels were separated, the former being discarded. We then randomly selected 16 movies from both the GFP and cherry expressing movies, after which a 200x200 pixel image slice focused on the right side of the primitive streak was taken from the middle frame of each of the selected time-lapses.

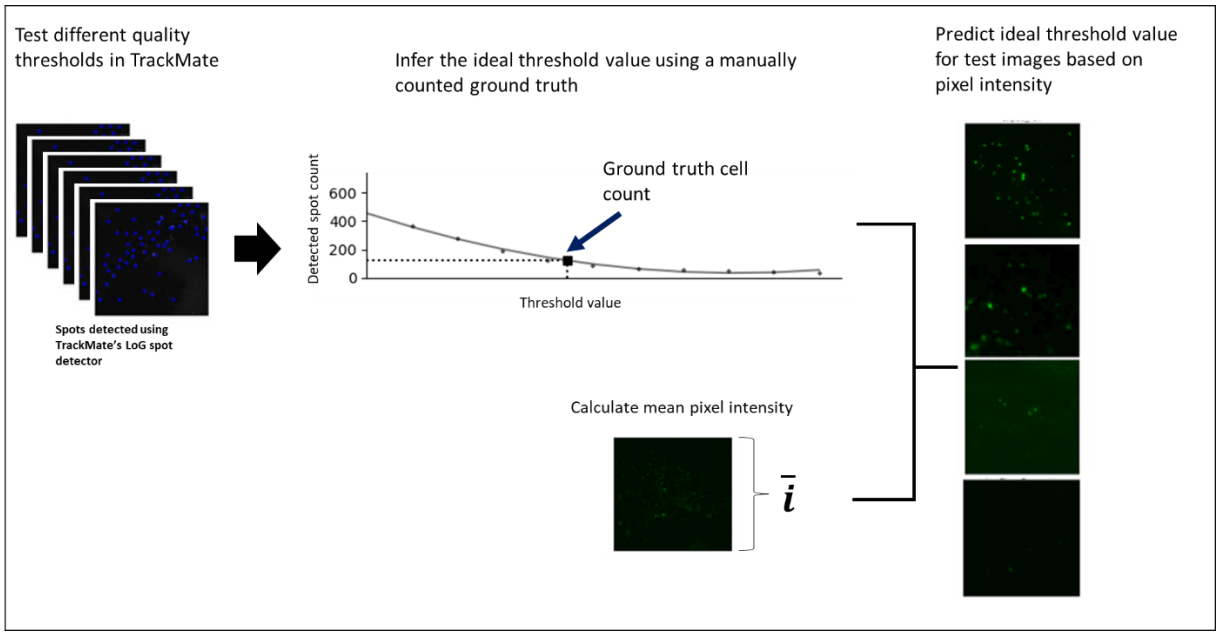


Figure 2.1 Methodology of generating a predictive model for estimating an ideal quality threshold in TrackMate's spot detector using the mean pixel intensity. The most accurate ideal quality threshold is found using a manually counted ground-truth, from which the ideal threshold is predicted in test image partitions.

An initial cell count was produced for each image slice using ImageJ's multi-point tool, which was used to represent a ground truth value to compare against numerous cell capture tools, including TrackMate, CellProfiler, SpotTracker and NucliTrack Testing showed CellProfiler consistently over-segmented the image, SpotTracker detected too few cells (<5), while NucliTrack crashed during the segmentation stage. Therefore, TrackMate was the only software considered suitable to be used for our TLM-movies.

We ran TrackMate's Spot Detector using the LoG method with the blob diameter set to 6 pixels and median filtering selected and repeated the detection using quality thresholds ranging 0.0 – 1.0 in intervals of 0.1 for each image slice. We used the cell detection counts across these thresholds to produce a simple 2nd degree polynomial linear regression model, following which the ground truth cell count was used to predict what the ideal quality threshold should be for each image.

As shown in Figure 2.1, we then calculated the mean pixel intensities for each image, which were used alongside the ideal quality threshold to produce a linear model for images including GFP, cherry and the combination of both fluorescent signals. We selected the model with the lowest p-value to predict the ideal quality threshold in any given image.

2.2.4 Testing TrackMate's single-particle-tracker

To investigate how successfully TrackMate's LAP algorithm successfully links individual cells across frames in our context, we tested the same 32 movies selected for cell detection, but this time used the full-time-lapses and did not crop the images. To select an appropriate quality threshold for each time-lapse, we first calculated the grand-mean pixel intensity, referring to the mean of mean pixel intensities across all frames, which was then used as an input for the linear model described in 2.2.3 to predict the ideal threshold value for the LoG Spot Detector. For the tracking method, we used the LAP Tracker with a

maximum distance of frame-to-frame linking set to 30 pixels with track segment gap closing set to a 30 pixel max distance and a max frame gap of 2, and as cells were not expected to divide or fuse together, we set track segment splitting/merging off.

In the absence of a ground-truth for cell linkages to identify specific errors in TrackMate's results, we chose instead to determine the difference between the number of cell detections included in tracks with the total number of tracks generated as a proxy for errors in frame-to-frame linkage and gap closing. The cell count was established by identifying the maximum number of cells in any frame, excluding those which were not considered part of any track.

2.2.5 Statistical analyses

Statistical tests were carried out using SciPy and Statsmodels in Python 3.7. We tested data for normality and equality of variances using the Shapiro-Wilk and Levene's test respectively. All comparisons performed between GFP and cherry expressing movies and other independent values were made using the Mann-Whitney test.

2.3 Results

2.3.1 Identifying optimal TrackMate parameters

We randomly selected 16 movies of migratory cells in early embryonic chick development, including those expressing GFP and cherry, and multiple experimental interventions such as overexpression of BMP2, FST, Wnt3a and Smad1, as well as morpholino knockdowns. We took the central frame from each movie and cropped each image to equal-sized partitions focused on the right cluster of cells as samples for testing TrackMate's cell detection, as shown in Figure 2.2 and S2.1-2.2.

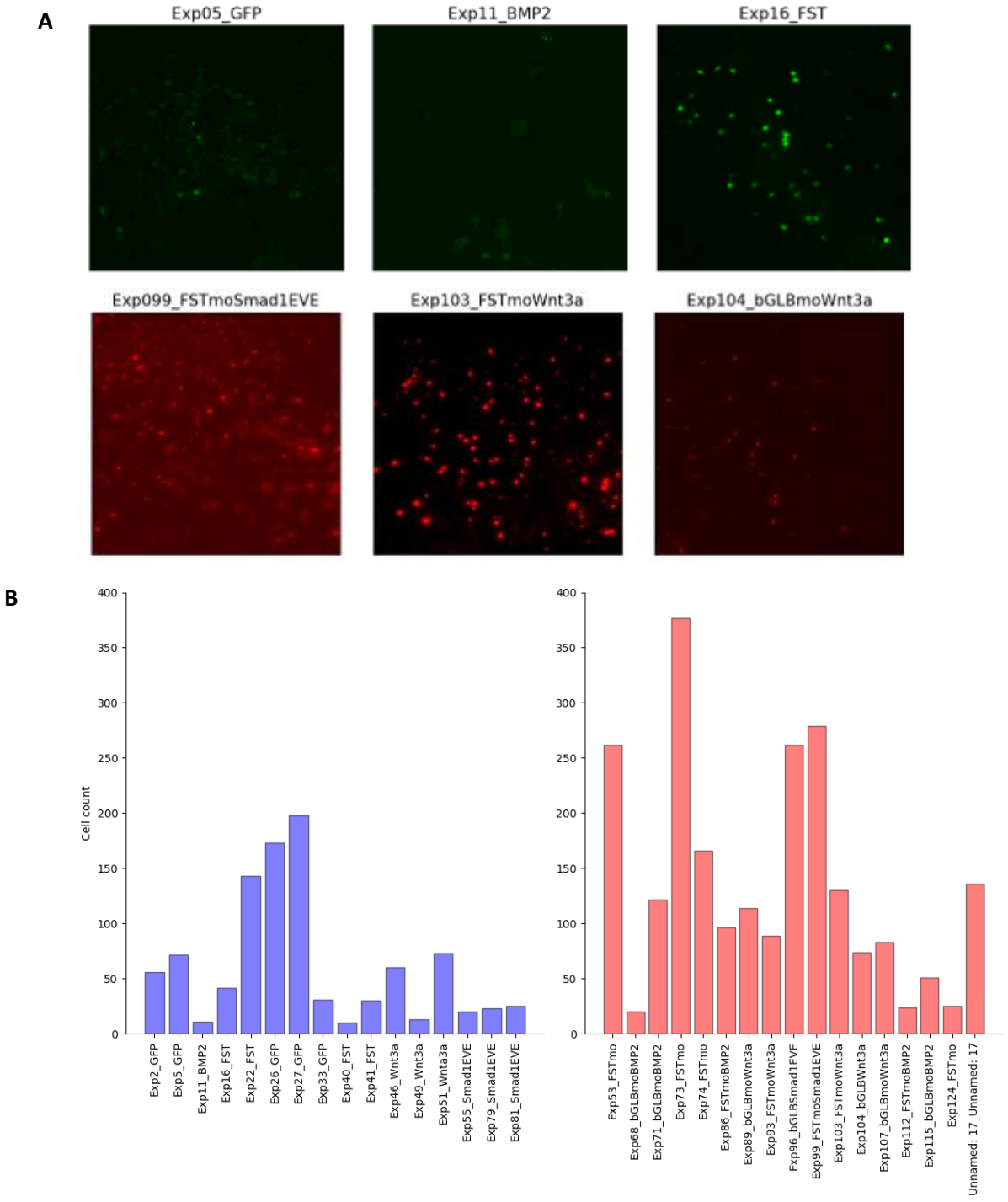


Figure 2.2 Exemplary 200x200 pixel image slices from GFP (top A) and cherry (bottom A) expressing cells in time-lapses of early embryonic chick development, with histograms (B) showing manual cell counts for all images shown for GFP (blue) and cherry (red).

Manual counts were made for each partition (Figure 2.2B), establishing a ground truth to compare with detections made using TrackMate's Spot Detector, or custom designed tools. Across all images, the mean cell count was 98.6 cells (SD 91.0), with 61.3 (SD 57.2) in GFP expressing cells and 135.9 (SD 102.4) in those expressing cherry, however since each image partition can be considered independent from one another, we did not compare means between groups.

We applied TrackMate's LoG spot detection algorithm for each image partition using sequential threshold values ranging from 0.0 – 1.0, the determined cell counts of which were used to produce a linear model that allowed us to input the partition's ground truth cell count to predict the ideal quality threshold value as shown in S2.3-2.4. The predicted ideal threshold values were not uniform across all partitions, with the value for combined groups being 0.83 (SD 0.41), and 0.91 (SD 0.45) in GFP and 0.73 (SD 0.31) in cherry expressing movies specifically, with no significance found between the two groups ($P > 0.05$).

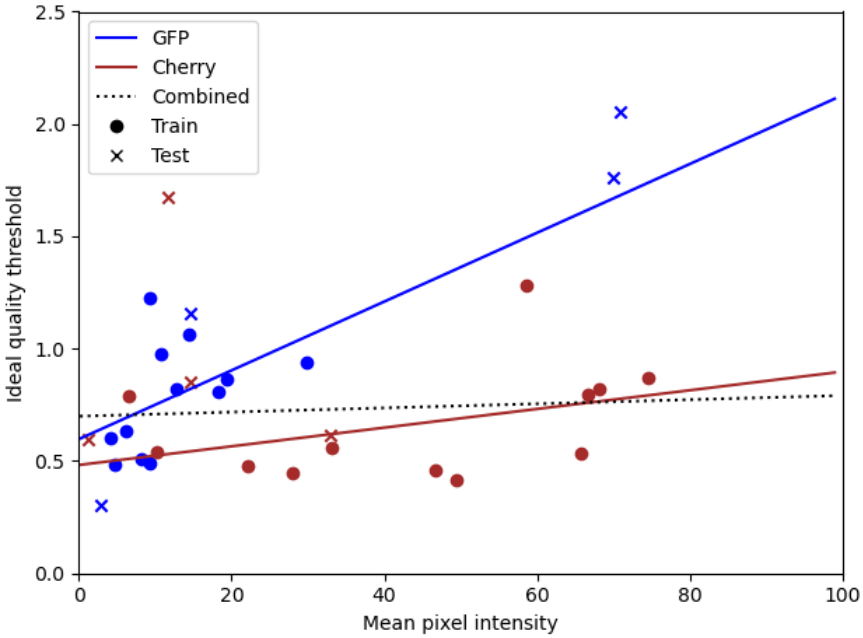


Figure 2.3 Predicted ideal quality threshold determined using polynomial regression TrackMate's LoG spot detector as the mean pixel intensity of an image changes, with linear regression models for GFP (blue), Cherry (red) and combined (black) fluorescent biomarker groups.

For each image partition, the mean pixel intensity was calculated, showing a grand-mean intensity of 27.98 (SD 24.01) in the combined groups with 19.09 (SD 20.50) in GFP images and 36.87 (SD 23.98) in those expressing cherry , the difference of which was shown to be significant ($P < 0.05$). We used the ideal quality threshold and mean intensity of training partitions to produce linear models in the combined groups and in each group separately (Figure 2.3). In our training set, the MAE was 0.162 in GFP images, 0.170 in cherry and 0.201 using a model combining both fluorescent tags. Under evaluation, GFP and cherry images separately gave lower MAEs at 0.284 and 0.392 respectively than the combined model at 0.557, thus we chose to use each model for their respective fluorescent label independently.

2.3.2 Evaluating trajectories generated in TrackMate

Having generated two models which can estimate a suitable quality threshold for TrackMate's spot detection algorithm in any image based on its mean pixel intensity, we next decided to test the tool's accuracy in generating cell tracks in full time-lapses of uncropped frames when segmentation is determined by a uniform function. Using the full and uncropped TLM-images from all the movies in which we obtained our 200x200 partitions, we first determined the grand-mean pixel intensity calculated from all frames from each time-lapse, which was then used to predict an ideal quality threshold for spot detection before initiating the cell-linkage algorithm.

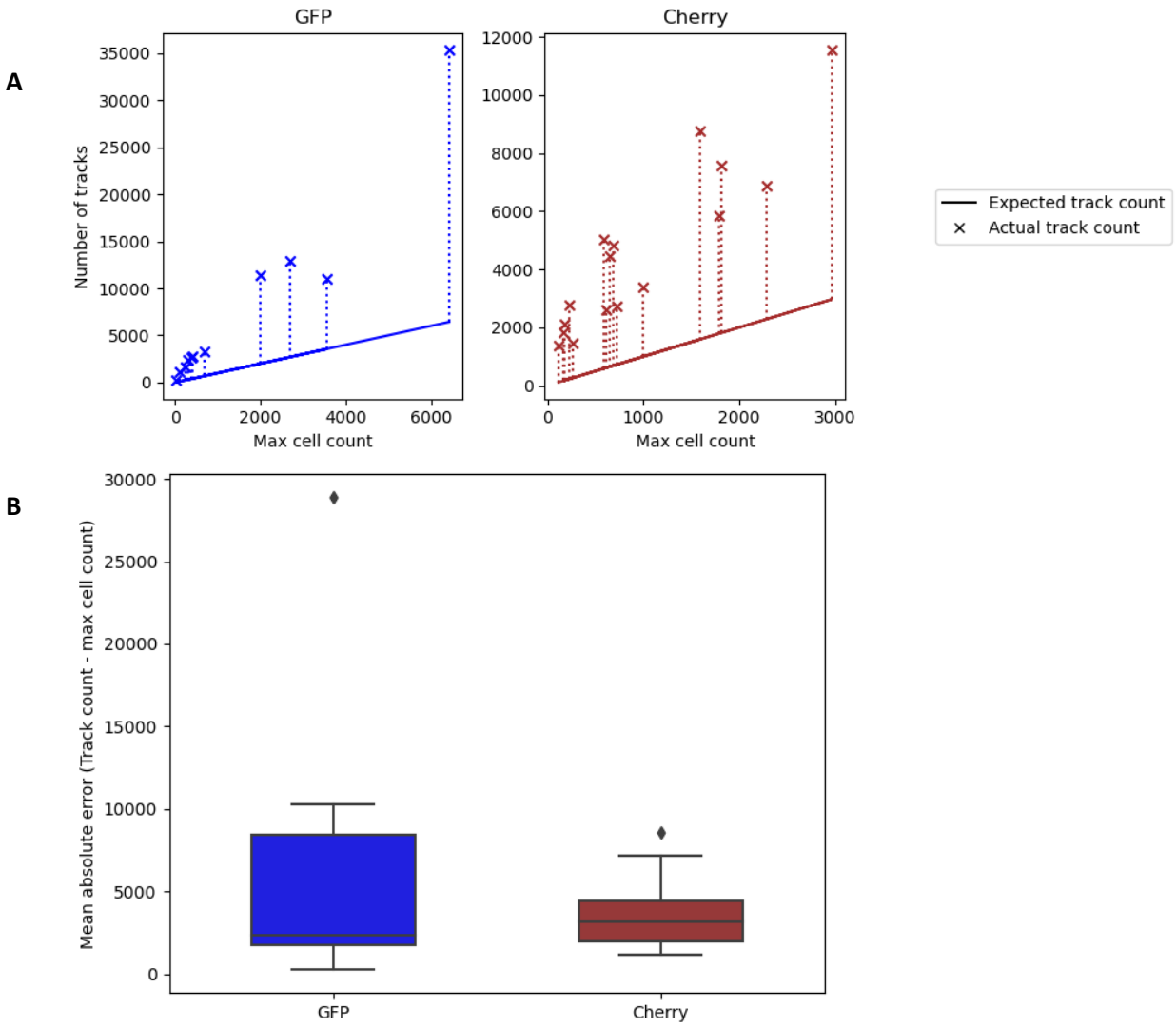


Figure 2.4 Difference in the maximum cell count in any given frame of the movie vs. the total number of tracks generated in TrackMate (A). The absolute errors between cell count and track number in GFP (blue) and cherry (red) movies (B).

We visualized the tracks generated by TrackMate in the TLM-movies, which appeared to contain numerous errors in the cell linkage stage, with segmentations of the tracks suggesting a failure in gap closing between frames and sharp angles in some trajectories indicating cells may have linked together incorrectly. As TrackMate identified well over 1000 tracks most movies, quantifying specific failures in the cell-linkage tool by manually interpreting the visualizations would be time-consuming and prone to error; we chose instead to use the difference between the number of cell detections and the number of tracks as a proxy for said linkage-errors as shown in Figure 2.3, as in an ideal analysis these values would be expected to be identical.

Across GFP and cherry movies combined, the mean maximum cell count was 1201.85 (SD 1407.46) and the mean number of tracks was shown to be 5865.63 (SD 6772.01), the difference of which was found to be significant ($P < 0.05$). The MAE between the maximum cell detections and the number of tracks was 4654.78 (SD 5446.68) in combined groups, with 6199.18 (SD 7902.66) in GFP and 3593.00 (SD 2087.82) in cherry expressing movies specifically, with no significant difference being found between the two groups ($P > 0.05$).

2.4 Discussion

While we showed TrackMate's LoG spot detection algorithm has the potential to capture a quantity of cells close or equal to the number of cells countable by the user in specific images, this is dependent on an optimal threshold value which appears to vary across different images. This inconsistency could be problematic for the reproducibility of analyses involving different images with varying features as well across frames of a TLM-movie, since an accurate cell detection in each image could require a unique threshold value that would be unknown to the user. As an example, a previous study investigating cell motion in somite morphogenesis [28] made no mention of the threshold value used and did not provide an approach for finding a suitable value, meaning other researchers could have difficulty reproducing the method.

Our decision to use a statistical approach to predict an ideal threshold value was driven by a desire to use a reproducible and justifiable method to capture cells across each TLM-movie, rather than merely estimating the value "by eye". The ideal threshold value generally increased proportional to the mean pixel intensity of an image with a relatively strong correlation, but since small changes to the threshold value can have a large impact on the cell detections, the model would not be expected to accurately predict an optimal value, though we did not have ground-truth data to test this. TrackMate also uses a single threshold value for the entire movie, meaning variations between image frames were not considered, resulting in us having to use a grand-mean of pixel intensities across all frames to predict the value, which may have been unsuitable for cell capture in some frames.

Our regression model used the mean pixel intensity as the only variable to estimate the ideal threshold value, however it would be possible to use a multiple regression analysis with other image features, such as median intensity, weighted average intensity and pixel variance, as well as using said values when images are processed through different convolutions or binarizations. It is possible this could result in a model which can predict the ideal threshold with more accuracy, however as previously stated, this would require more image datasets for a more valid evaluation of the model's performance.

Another possible limitation in the reproducibility of the particle detector's algorithm for capturing cells is that the cell diameter must be user-defined, which TrackMate's user manual itself identifies as an area for future improvement [41]. As the size of the Laplacian kernel, which is used in segmenting the objects and assigning a quality value, is determined by the diameter, cells smaller or closer than this constant value could be falsely removed by the threshold, while larger cells may be over-segmented into multiple false detections. Since we did not consider the size of individual cells or validate the accuracy of the detections however, we cannot say how this limitation affected our results, but if an approach was used to identify whole objects through contour modelling [57], we could be able to detect cells of different sizes more effectively.

To test TrackMate's spot linkage algorithm, an exhaustive approach would have involved manually tracking each cell and comparing it to the tool's results [31]. We were unable to use this method, as manually tracking cells with high density and hundreds of frames would be too time-consuming and prone to error. We instead chose to use the difference between the number of tracks generated and the maximum number of cells detectable across all frames of a movie as a proxy to represent errors such as mis-linkages and failed gap closing events. An idealised and error-free cell tracking analysis would be expected to generate a number of tracks equal to the number of detected cells, which is in contrast to our analysis that showed that the number of tracks was significantly larger than the maximum detectable number of cells across all movies. This indicated that even with consistent estimations for spot detection, numerous errors in cell linkage were occurring.

It was also notable that the difference between cell and track counts increased significantly as the cell count increased, suggesting that the tracking could be improved using a smaller number of high confidence cell detections by increasing the quality threshold. This would however mean a large amount of movement information is lost, with high confidence cells potentially being a poor representation of the overall CPC population. Our analysis also showed that even when the cell count was relatively small, the track number was still considerably greater than the cell count, suggesting that the high cell density alone may not be enough to explain the errors in our results.

In previous studies where TrackMate appears to generate relatively linear and connected trajectories, the analyses tend to use data with features that would be expected to reduce the number of linkage errors, such as being high resolution in both imaging and time-pointing, containing low object density and involving cells which move linearly with minimal stochasticity [28, 52–54]. This is in contrast with our own TLM-images, in which cells are not highly resolved, they are often highly abundant with up to 1000 possible detections in a single frame, and their movement is sporadic with frequent occlusion events, suggesting TrackMate's method may be unsuitable for our analysis.

As the chick emrbyo is a relatively rare context where we can assume there will be two main groups of cells which emerge from the primitve streak and migrate bilaterally, we considerd tracking the movement of these cell groups as a whole rather than tracking each cell indivually, which would avoid many of the issues found in single-particle-tracking such as mis-linkages and gaps. Many investigations have previously tracked the movement of cells during early chick emrbyos which did not involve single-particle-tracking, but instead generated trajectories based on the combined movement of multiple signals, either through the successive additon of frames [5, 6, 18] or by using optical flow-based methods to estimate cell or tissue velocity [58]. A possible disadvatnage of these methods is that they do not involve detecting cells as objects but instead follow their singals, which could vary between cells and be an inncaruate representation of their density. It may be possible to overcome this limitation by using cell detections first before assinging them into separate groups and tracking them based on their density across frames using a mean-shift approach [59].

Chapter 3. Improved detection of cardiac progenitor cells

To analyse the migration of CPCs in our TLM-movies, we wanted to use a method in which detections are an accurate representation of the CPC population and allows for the largest possible number of true cells to be used in generating trajectories with minimal false detections. The results from Chapter 3 indicated that while TrackMate can be optimised to produce accurate cell counts in our TLM-movie partitions, the ideal parameters were unique for each image. This could lead to issues in reproducing the methods to capture CPCs in future investigations, as researchers may be left to estimate these parameters “by eye” or instead rely on manual counting.

Manually counting cells in our images would be exhaustive and time-consuming due to the large number of frames in our TLMs. While we attempted to overcome the reproducibility problem using a regression model to predict an ideal quality threshold based on an image’s mean intensity, the fit was not strong enough to assume the estimation could be used to accurately capture cells across diverse images. This also did not resolve other discussed issues, including parameters being non-adaptive across time-lapsed frames and a detection approach that assumes uniformity in cell sizes in each movie.

For singe-particle-tracking tools such as TrackMate, an accurate cell tracking would necessarily follow an accurate detection of said cells, as demonstrated by the tracking algorithms which participated in the Cell/Particle Tracking Challenges [31, 32], where the most accurate cell tracks were consistently followed by the most accurate cell detections. It is possible that TrackMate’s failure to generate accurate and consolidated tracks in our movies was a result of an inaccurate cell detection stage, which may itself be due to the fact our movies contain dense frames with cluttered cells that appear to express their signals transiently, since they are often visible in some frames but disappear in others. This issue could be avoided by detecting only high confidence spots after selecting a higher quality threshold, but this may come at the cost of analysing a sample that is not representative of the entire cell population in terms of migratory behaviour.

TrackMate's spot detection algorithm is mathematically complex and designed to be robust in that it can be used across diverse biological contexts using different microscopy techniques, as evidenced by previous studies focused on the movement of cells or molecules [28, 52–54]. As we were unable to reproduce accurate cell detections and trajectories similar to these investigations however, it may be a suboptimal tool for specific scenarios such as in capturing CPCs in fluorescent microscopy images in chick embryos, suggesting an alternative approach may be required.

It is possible to abandon the conventional single-particle-tracking approach of detecting and linking individual cells by instead tracking the change of pixel intensities using an optical flow approach [58] or by sequentially imposing successive frames to produce a track [5, 6, 18], leading to the generation of an overall trajectory of changing intensities. While tracking cells as a combined group may be useful in avoiding cell linkage errors resulting from cluttered and transient signals, generating a trajectory based on estimated pixel velocities could lead to larger cells or those with greater signal expressions being overrepresented when analysing cell migrations.

We sought to develop a custom segmentation and cell feature extraction scheme developed specifically for detecting fluorescently labelled CPCs in microscopy movies of chick embryos. With our fixed biological context, we wanted to explore and evaluate classical computer vision techniques used for object detection as well as to test their accuracy against the more complex TrackMate spot detection algorithm.

3.1 Cell Segmentation

Image segmentation is a form of image processing that refers to the separation of sets of pixels in an image into simplified homogenous regions, where objects-of-interest of the foreground are distinguished from the background [25]. Image segmentation is critical for detecting objects-of-interest as well as characterising and extracting their features such as quantity, size, shape and positions. In unprocessed images the intensities of both the foreground and background are not uniform and the object boundaries are undefined. This means extracting said features directly would be challenging even in manual detection and implausible using algorithmic methods.

A segmentation algorithm designed for cell detection can be judged based on its power of capturing true cells (or true positives) and its tendency to capture false cells, also known as the false discovery rate (FDR). A perfect segmentation of an image with ideal quality would be sensitive enough to allow the identification of all true cells while also being specific to said cells with a FDR of zero, but this is rarely achievable due to limitations in current segmentation methods and imaging techniques, as cells may be lost in some frames or falsely detected due to low SNR, uneven lighting and overlapping of cells.

For cell tracking tools, it is essential that the segmentation method allows for a sensitive and specific detection of cells, as the proceeding cell-linkage algorithm cannot perform accurately when true cells are undetected or false cells are detected. The importance of segmentation is supported by the *Ulman et al.* review [31], as for most of the CTC TLM-movies, the methods with the top cell-linkage performances also used the top performing segmentation algorithms.

In most current automated tracking methods, the segmentation algorithms used to capture cells follow a generalized principle shown in Figure 3, where the images are first pre-processed using filters, usually to remove noise that could falsely be captured as cells. The main segmentation processing step is used to detect the signals, but the algorithms can involve various combinations of approaches including using additional filters to detect particle-like "blobs" or edges [41, 60], binarizing the image into simplified regions [61], and transforming the image using morphological operations to ensure cells are presented as single objects [62]. The position of each cell is typically found by determining their features, such as centroids, or through detecting peak intensity maxima. Post-processing may also be used to filter non-cell signals based on features such as size or shape.

Cell segmentation using sophisticated machine learning techniques and convolutional neural networks can perform highly competitively compared to the more standard approaches, but their development is complex and requires a large amount of data to train [63]. This means that, despite their increasing in popularity, most current methods are developed with more basic principles.

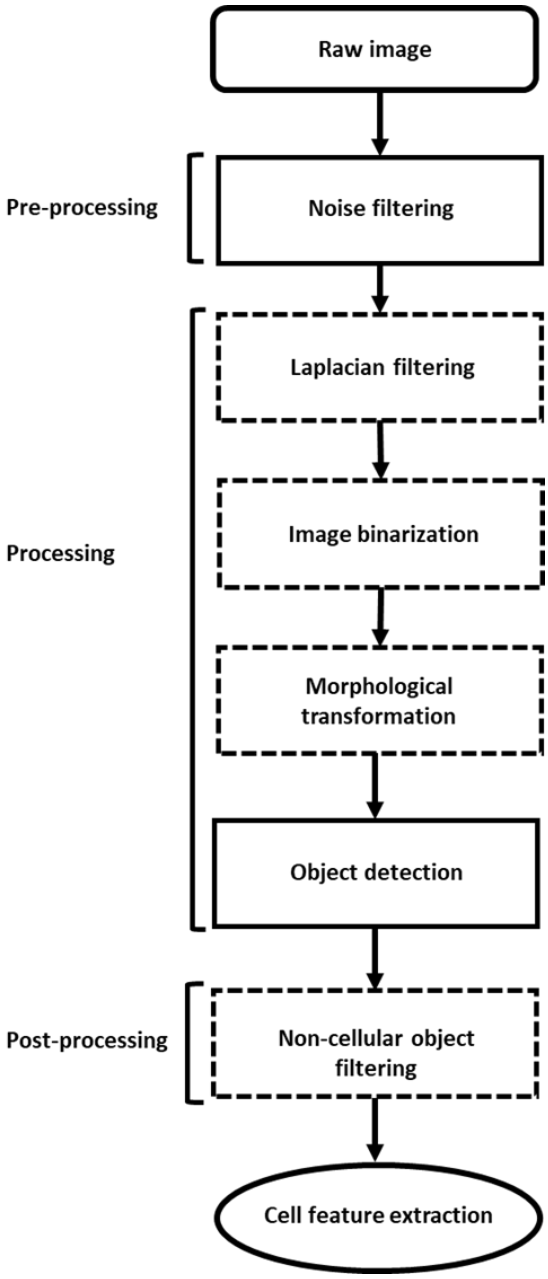


Figure 3.1 General approach for segmentation in cell tracking methods, involving pre-processing to remove noise, main processing where cells are detected and post-processing to further filter noise. Dotted boxes represent stages which are common, but not universal across all segmentation algorithms. Adapted from *Hamilton, 2009.*

Cell segmentation methods can vary greatly in sensitivity and specificity when capturing cells in different types of images, with algorithm reviews for both the CTC and PTC suggesting there is no single approach that performs optimally across all images produced using different microscopy techniques or within different biological contexts [31, 32]. In the case of images with fluorescently labelled particle-like cells, features such as intense background noise defined by the signal-noise-ratio (SNR) and uneven lighting are common which can demand more complicated filtering or thresholding methods compared to DIC or PC images [26]. The most widely used cell tracking software are applicable to many image types and biological contexts, usually involving semi-automated segmentations, which is likely why they are so versatile. A method designed specifically for detecting particle-like fluorescently labelled cells in TLM-images of whole embryos could potentially outperform current these software in said context however.

3.1.1 Image convolution

Convolving an image refers to a matrix operation that uses miniature matrices called kernels or convolution matrices, which can consist of different shapes, sizes and arrangements of values that act as weights. The kernel centers on each image pixel and determines the weighted sum of all local pixel intensities that fit within the kernel shape, which is then assigned said pixel [64]. The technique allows the filtering of specific pixel intensities from an image and can be used in cell segmentation approaches for removing noise, highlighting edges or detecting blobs.

(i) Noise filtering

The pre-processing stages of cell tracking algorithms usually involve noise removal, which is essential to prevent the detection of false cells, especially in fluorescent microscopy images, which often possess low SNRs compared to other imaging techniques. Image convolution is often used to smoothen or blur the images with the aim to reduce background regions with high intensity, which are assumed to be small points typically less than a few pixels in size.

Simple filters can use box blurs (figure 3.2A), where the kernel filter is designed with uniform values multiplied by a set fraction of one, but this method can lead to edges being too smooth to detect through the main segmentation process. In Gaussian blurring, shown in Figure 3.2B, the kernel values are produced by a Gaussian function, which can be used to filter noise while also retaining distinct edges [65], making it preferable for cell detection. Some techniques can result in even more edge retention, such as median filtering [66].

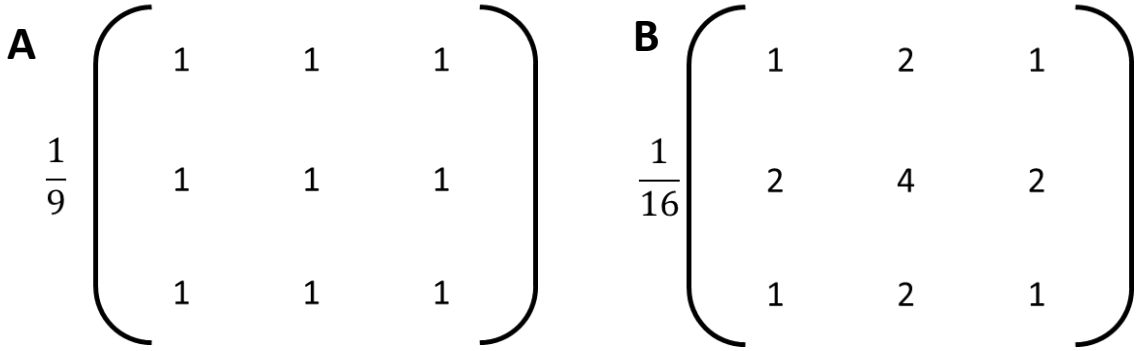


Figure 3.2. Kernels designed for noise filtering including a basic box filter (A) and a Gaussian filter (B).

(ii) Blob detection

For particle-like cells, image convolution can highlight them as small regions of high intensity against the background, after which they can be detected by thresholding and labelling, or by finding peak intensities across the image. For this purpose, Laplacian kernels are often used (Figure 3.3), which are distinguished by containing large values at the center and much smaller (often negative) values at the peripheral, or vice versa. Kernels can be automatically produced with such characteristics by applying a Laplacian operation on a Gaussian distribution, resulting in a Laplacian of Gaussian (LoG) filter, which is used in the TrackMate spot detector [67]. A Difference of Gaussian (DoG) approach, where the difference of an image is found when applied with two Gaussian filters, can offer similar results to using LoG filters.

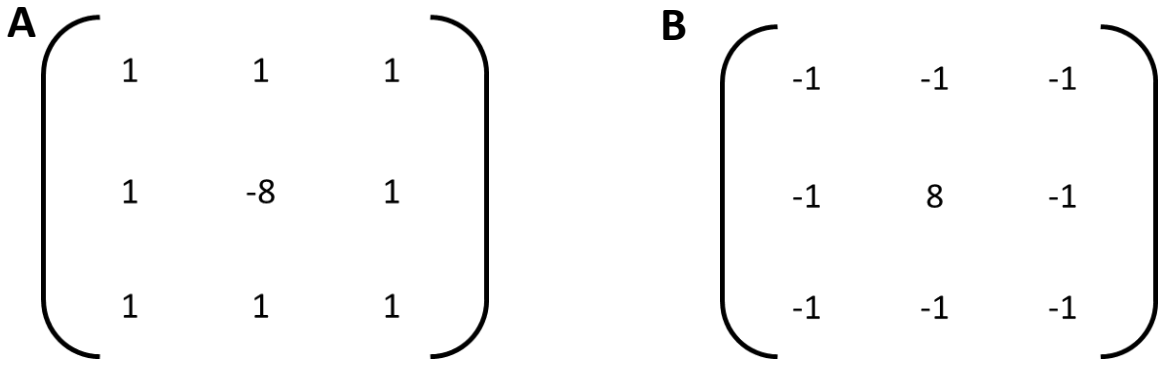


Figure 3.3 Laplacian kernels designed for blob detection for low intensity blobs (A) and high intesity blobs (B).

3.1.2 Thresholding

Thresholding is one of the most common segmentation techniques and involves categorizing pixels in an image into those with intensities that are above or below specified values, allowing the image to be partitioned into distinct regions [25]. Typically, a single threshold value is used to binarize the image, which, assuming the intensities of the foreground are sufficiently distinct from those in the background, allows for objects-of-interest such as cells to be separated from the background and detected.

In greyscale images where the pixel-intensity histogram displays two or more peaks a global thresholding approach, where the threshold value is the same for all pixels, is often used to detect cells. While the threshold value can be set manually, finding the value that allows for the greatest cell detection specificity and sensitivity can be time-consuming, especially when analyzing a time-lapsed series of images, where the ideal threshold value may vary between frames. For this reason, automated methods are typically used to predict the ideal threshold value of an image based on the distribution of intensities. Techniques involving blob detection often use an automated or manual global threshold to homogenize the high-intensity regions of cells following their convolution.

Otsu's thresholding was one of the earliest automated global techniques developed for greyscale image segmentation [68], ideally used when pixels display a bimodal distribution. Based on the assumption that the image foreground and background form each histogram peak as two separate classes, the algorithm finds the intensity with the maximum variance between classes and assigns it as the threshold value. Previous studies have successfully extracted cell features following the use of Otsu's thresholding in DIC/PC microscopy images or in fluorescent images with low background noise [26, 69], as these images fit the assumptions of a bimodal distribution of pixel intensity. Microscopy images of fluorescently labelled cells are however often saturated with background noise that may lead to a pixel-intensity histogram that is unimodal, which is common in fluorescent imaging as a result of uneven lighting caused by background autofluorescence, overexpression of fluorescent proteins and photobleaching [61]. In such images, Otsu's method and other global thresholding approaches are considered unsuitable as the background noise can be falsely detected as cells.

K-means clustering is an unsupervised machine learning approach that is used across a wide variety of different contexts to establish different classes based on the structure of the data, but is similar to Otsu's method in that it exhaustively searches to find the point of maximum variance between classes and assesses each pixel by a global standard. K-means is generally considered more computationally efficient in image binarization than Otsu's method [68], making it an important algorithm to compare with, but our segmentation scheme would not benefit from its other advantages such as allowing for more than two k-numbers and the segmentation of colour images.

To resolve the segmentation issues found in images with unimodal pixel intensities, adaptive thresholding techniques (or dynamic thresholding) are often used, where a mutable threshold value is assigned to each pixel as opposed to all pixels in global techniques. Each adaptive threshold value is based on the intensities of local pixels, with the most common techniques including adaptive mean thresholding, where the threshold value is based on the mean of the local area, and adaptive Gaussian thresholding, where the value is based on a Gaussian weighted sum of the local pixel intensities.

There is however a tendency for fluorescent images to be over-segmented by local adaptive thresholding techniques, where the cells are often hollowed and fragmented into multiple spots, making it difficult to detect each cell. Morphological operations such as closing, where images are sequentially dilated and eroded to combine the disjointed fragments into whole cells, are often used in an attempt to correct this issue [70], but this itself could lead to issues where smaller cells could be conjoined to other cells or lost by the transformation, especially in images with particle-like cells.

3.1.3 Morphological operations

Mathematical morphology refers to operations that can transform binary images, the most basic operations of which include erosion and dilation. The operations work similarly to convolutional filtering, but instead applies a kernel with a specific shape called the structuring element (Figure 3.4) across each pixel [71]. In erosion, if any pixels within the structuring element are not equal to 1, said pixel will be assigned 0, leading to a portion of the object boundary being removed. In contrast, the dilation operation assigns 1 to the pixel if at least one other local pixel is also 1, causing the boundaries to expand. The size and shape that a structuring element takes within a kernel has a significant impact on the result of the morphological operation, with those used on cells typically being elliptical due to their shape typically being circular.

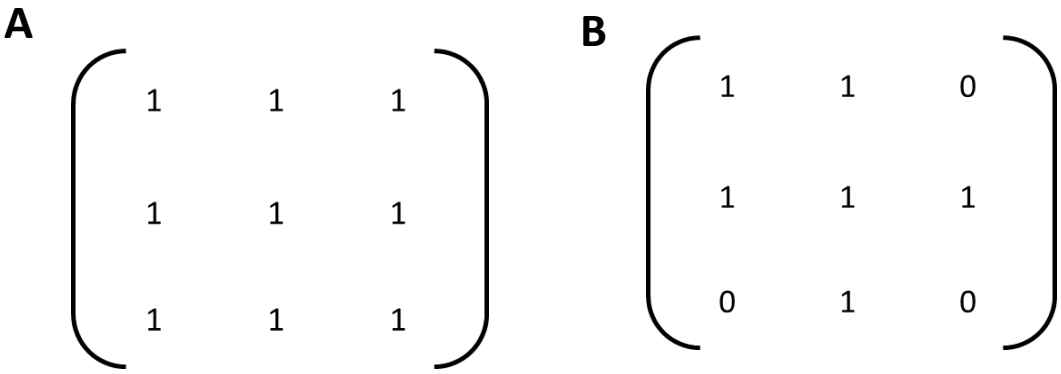


Figure 3.4. Kernels with structuring elements containing rectangles (A) and ellipses (B).

(i) Opening and closing

Opening and closing combines erosion and dilation operations by performing them consecutively. These are common techniques used in the post-processing stages of cell detection following segmentation and binarization. Opening involves erosion followed by dilation, resulting in objects smaller than the structuring element being removed while mostly respecting the original size of larger objects which can be useful in determining cell boundaries [72]. While opening has been used as a noise filter in some contexts (Argawai and Doermann, 2009), for filtering in cell images it could lead to a loss of lowly resolved particle-like cells. In contrast, closing involves image dilation followed by erosion, often leading to the joining of previously separate objects or the filling in of hollow objects. The technique has been used to de-hollow and reform single cells following their over-segmentation by adaptive thresholding [70], but it risks closely clustered cells becoming joined.

(ii) Watershed

Watershed transforms are commonly used in image segmentation to define boundaries of cells that overlap, which would make them indistinguishable in contour detection using binary thresholding alone, as well as labelling them [73]. The algorithm is based on elements from topography, where the cells are considered points of low elevation with high elevation boundaries, after which cells are "flooded" as separate regions. Many watershed algorithms are based on an initial thresholded image that is distance transformed. Opening and closing techniques are used to define markers located roughly at the centre of each cell before being used as points of "flooding" [74], thus allowing the boundaries of cells to be distinguished.

3.1.4 Feature extraction

Detection association-based cell tracking algorithms require features of cells to be detected across frames, with the most universal being position but can also include intensity, size and spatial features, as reviewed by *Meijering et al.* [75]. For tracking particle-like cells of near-uniform size, the position and intensity are often the only features used, since it is impossible to distinguish other features. Detection usually involves finding a centroid to each cell, which can occur following Laplacian filtering, thresholding, watershed transforming, or a combination of the three.

(i) Extrema detection

Extrema detection is a method of detecting cells used in TrackMate [67], which is most effective following segmentation processes that involve Laplacian filtering or other blob detection approaches. Particle-like cells can be detected as described in Chapter 2- by applying a LoG filter, from which sharp local maxima of intensity are detected before each pixel around each maxima are compared to local neighbours within a defined area, where the highest intensities are determined as the centre of the cell [55]. While the method is computationally efficient compared to contour detection, our results in Chapter 4 suggested it is difficult to identify optimal parameters that generalize across unseen datasets.

(ii) Contour detection

A common method of extracting cell centroid positions in binarized images involves first tracing the boundaries of an object using contours, where an active model is fitted across a boundary based the energinicity of the edge and the surrounding regions [57]. As opposed to using extrema detection, which could result in cells of different sizes being lost or lead to false detections, contour fitting is a more robust approach to cell detection that makes no prior assumptions about the cell sizes and allows the extraction of multiple cell features including centroid position, intensity, size and shape [76]. A disadvantage is that the method could become computationally costly as cell populations increase.

It is possible to use the region within contours to determine the central position, or centroid, of an object using image moments. Moments can be used to calculate the cartesian coordinates of a centroid in a binarized image via the formula:

$$x, y = \frac{\mu\ 1, 0}{\mu\ 0, 0}, \frac{\mu\ 0, 1}{\mu\ 0, 0}$$

Where the zeroth order moment $\mu\ 0, 0$ is used to sum all interesting pixels (non-zero) within the contour, which can be used to represent the object's area, using the formula:

$$\mu\ 0, 0 = \sum_{x=0}^{w} \sum_{y=0}^{h} f(x, y)$$

The other moments $\mu\ 1, 0$ and $\mu\ 0, 1$ are calculated by determining the sum of all *x* and *y* coordinates respectively in which an interesting pixel is present using the formulas:

$$\mu\ 1, 0 = sum_x = \sum \sum x\ f(x, y)$$
$$\mu\ 0, 1 = sum_y = \sum \sum y\ f(x, y)$$

Contour fitting is most effective following a binary segmentation, as this would allow clearly defined edges to separate cells from background regions.

3.2 Cell detection methods

3.3.1 Image selection

We chose to analyse the same 200x200 pixel image partitions used to generate the statistical model for predicting an ideal quality threshold described in Chapter 2, but only included GFP expressing movies and excluded those taken from movies in which spot linkage in TrackMate did not complete. Cherry time-lapses were also excluded, as cells in these movies tended to shift in the opposite direction to which would be expected, even following image stabilization.

3.3.2 Classical segmentation methods

The OpenCV library [50] in Python 3.7 was primarily used to apply different computer vision techniques in our images, which were read in greyscale and converted to a Numpy array. We applied a number of classical segmentation techniques, including Otsu's binarization and k-means segmentation, which were run with automatic parameters. We additionally used local mean thresholding, local Gaussian thresholding and LoG filtering segmentation each with a pixel kernel size of 3x3. A marker-based Watershed transformation was then applied to the binarized images to establish small markers near the centre of each cell. An outer boundary for each cell was estimated by dilating the image and subtracting the markers, which was applied in black to define cells as independent objects.

To extract features from our partitions, we first fitted contours to the binarized images using the OpenCV findContours function to identify the locations of the cell boundaries. We then determined the cartesian coordinates of the centroid of each cell using an image moments function. We nested cell centroids for each image analysed into 2D arrays to allow feature extraction across multiple frames in further analyses. Cell counts were established for using different segmentation techniques by determining the length of the centroid array in each image, which were then compared with the ground truth and the TrackMate spot detection results using our linear model established in Chapter 3.

3.3.3 Custom convolution-based cell detection scheme

For our Laplacian inspired segmentation approach, we first generated multiple filters including those with kernel sizes of 3x3, 5x5, 7x7 and 9x9 and all elements, except for the central value, defined as -1, with an example shown in Figure 3.5. For each kernel size we generated unique filters with different central element to values, which ranged from 5 – 100 in intervals of 1. We then convolved the image with each filter and applied Otsu's binarization followed by a Watershed transform as described previously.

For our custom segmentation scheme, we detected cell centroids as described previously, but removed those with cartesian coordinates of [0, 0], which not considered true cell detections and should not be used for comparison to the cell counts determined by the ground truth and other techniques.

Code for the custom convolutional-based cell detector can be accessed:

https://github.com/ConnorReynoldsUK/MScRes_thesis/blob/main/Chapter3_SEGMENTATION.ipynb

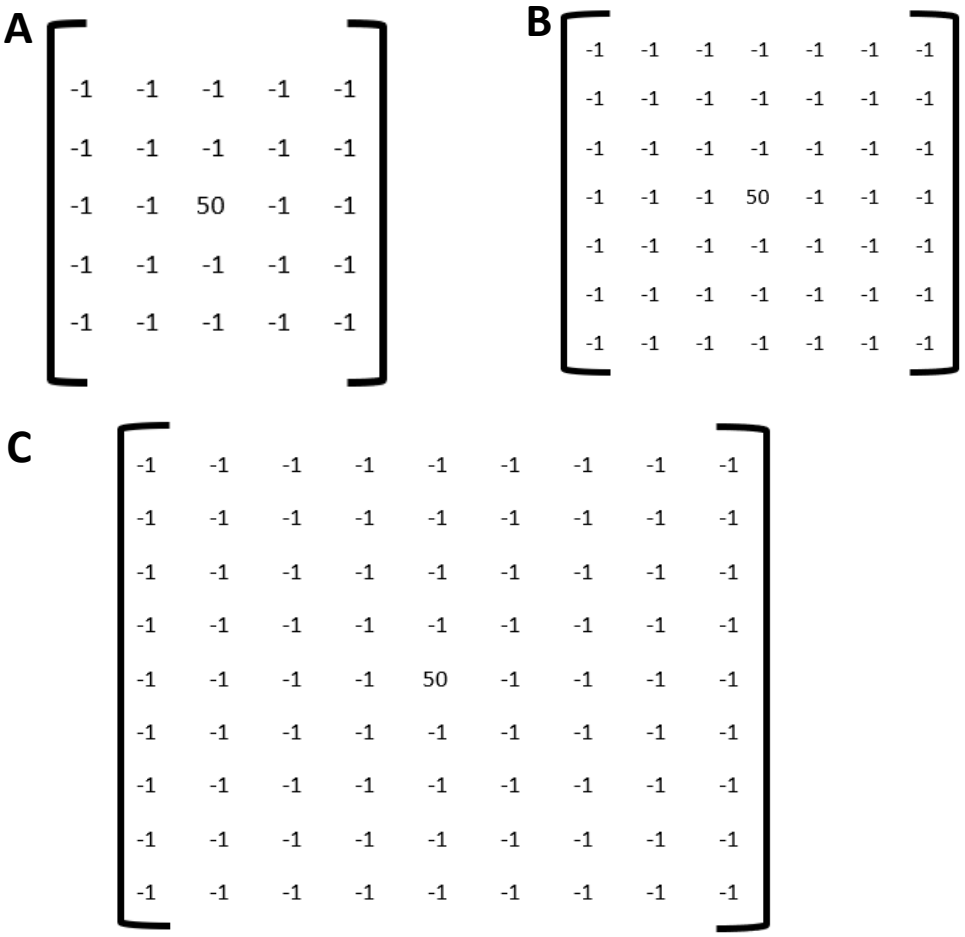


Figure 3.5. Custom Laplacian-inspired convolution filters developed each containing a central element value of 50 with kernel size 5x5 (A), 7x7 (B) and 9x9 (C).

3.3 Results

3.3.1 Classical segmentation methods

We ran each segmentation and first looked at the distribution of pixel intensities for each image partition prior and following each segmentation (Appendix A), which showed in all cases our raw images displayed a unimodal distribution. This initially suggested a global thresholding approach as used by Otsu or K-means would not be expected to be suitable for our images, since both approaches involve identifying differences within a bimodal distribution. The segmented images all displayed two distinct classes of pixel intensity, with a small number of pixels at 255, representing our objects-of-interest and a larger number valued 0, representing the background, thus confirming a binarization had taken place. While there were differences in objects-of-interest pixel numbers between each segmentation approach, we did not investigate these as they may not be representative of the cell count.

We interpreted our image partitions (Figure 3.6), where Otsu's and k-means segmentation visually appeared to produce similar binarizations in our partitions, with cells generally materializing as clear spots which sometimes clustered together, even after the Watershed transform. Both local thresholding approaches appeared to over-segment our partitions, resulting in noisy images with cells that seemed fragmented and were less distinguishable than shown following Otsu and k-means, which was surprising since global approaches generally perform poorer in images with a unimodal pixel intensity. The LoG approach was similar to local thresholding, but appeared less fragmented and contained a large number of connected cell edges rather than spots.

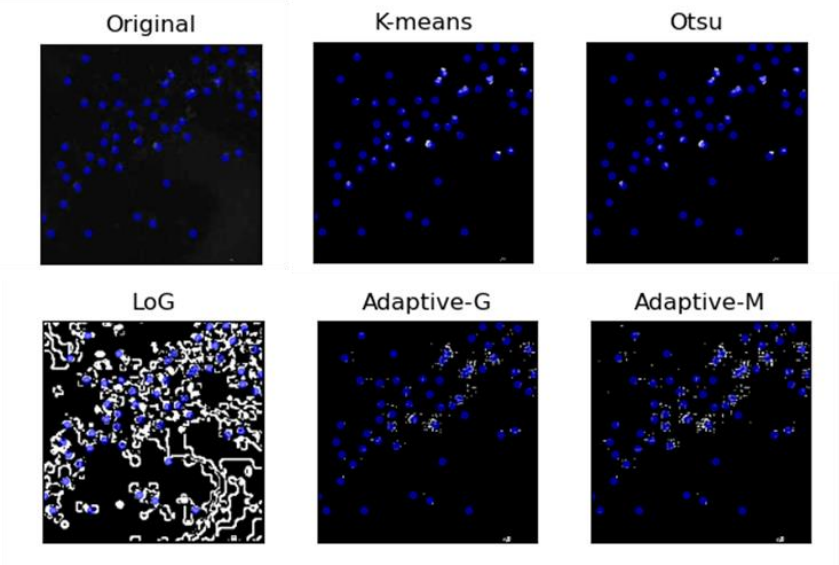


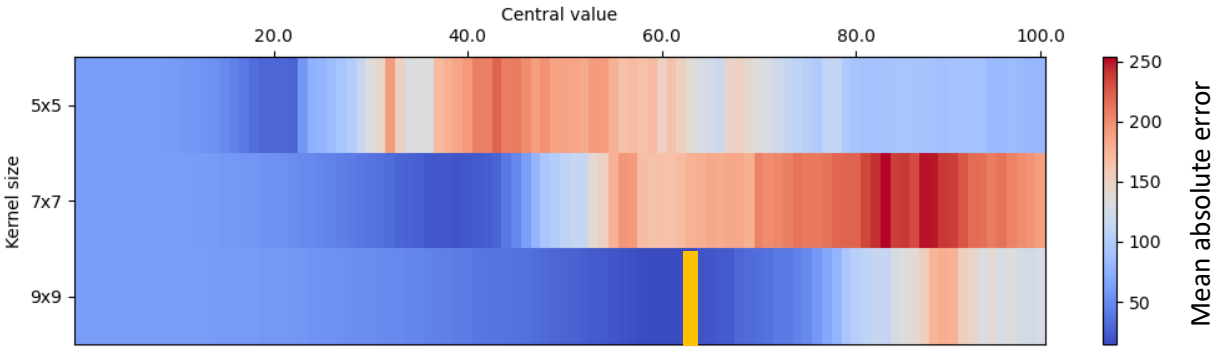
Figure 3.6 Image partition from Exp2_GFP including the original unprocessed image and those segmented by K-means, Otsu, Laplacian of Gaussian, Adaptive Gaussian thresholding and Adaptive median thresholding, in addition to the Watershed transform. Ground-truth sites displayed in blue.

The mean cell counts detected following said segmentation schemes showed that the k-means clustering approach had the smallest number of detections at 71.21 (SD 79.29) cells, closely followed by Otsu's binarization with a detection count of 71.43 (SD 78.36), supporting the previous suggestion that Otsu's and k-mean segmentations were similar based on the visual results. Larger values were found from the LoG method which gave a cell count of 396.57 (SD 240.48), in local Gaussian thresholding with 565.21 (SD 474.24) and local median thresholding with 510.14 (SD 451.84).

3.3.2 Custom segmentation scheme

We wanted to test an alternative approach involving image convolution using a custom Laplacian-inspired filter with a reduced central value in a spot detection-based segmentation for cell feature extraction. We first sought to identify a filter with the ideal kernel size and central element value for our image partitions by applying all combinations to find the convolution filter which gave the smallest difference between the mean detection count and the mean ground truth cell count (Figure 3.7).

A



B

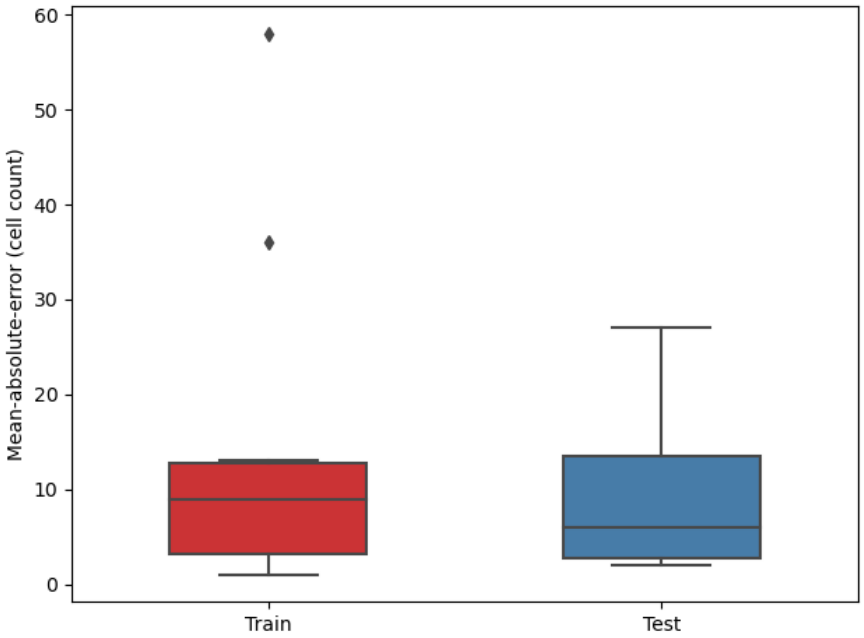


Figure 3.7 (A) Difference matrix showing the differences between mean ground truth cell counts and the number of cell detections made in GFP partitions with changing kernel sizes and central element values of a custom Laplacian-inspired filter. The smallest of 0.57 difference identified in yellow at kernel size 9x9 and central value 62. Colour bar represents the difference with red being higher and blue being closer to zero. (B) Mean absolute errors in Training and test datasets.

The optimal kernel filter was sized 9x9 with a central element value of 62, which, for our training dataset, gave the smallest MAE of 14.7 cells (SD 17.4) as shown in Figure 3.7. We ran this on our independent test set, which gave a MAE of 10.3 cells (SD 10.0). We considered this an appropriate error and thus selected this filter for comparison with other methods.

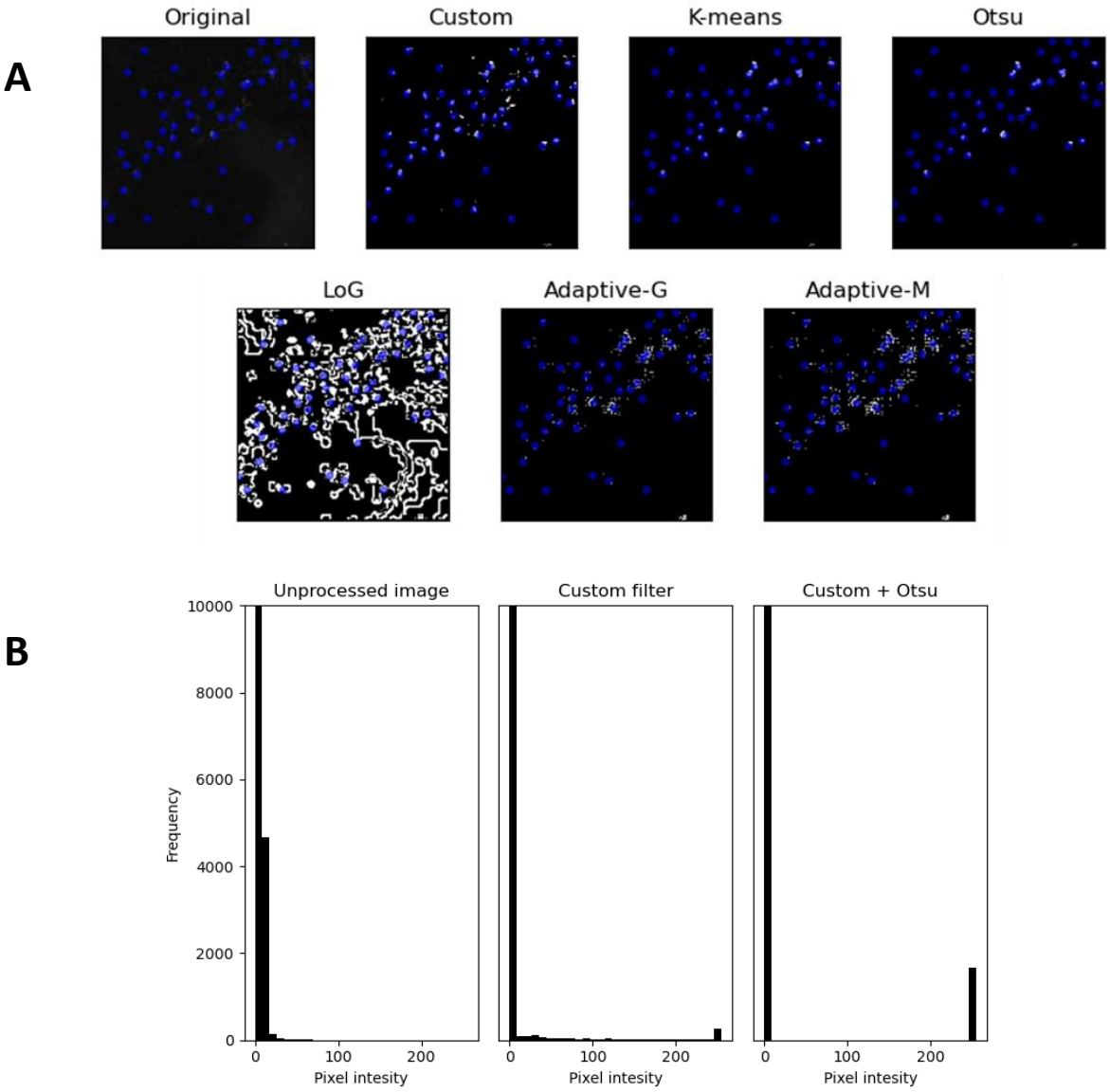


Figure 3.8 (A) Image partition from Exp2_GFP including the original unprocessed image and those segmented by our custom Laplacian-inspired filter, K-means, Otsu, Laplacian of Gaussian, Adaptive Gaussian thresholding and Adaptive median thresholding, in addition to the Watershed transform. Ground-truth points shown in blue. (B) Histograms showing the distribution of pixel intensities in Exp2_GFP in the original partition, the image following segmentation by our custom filter and the final segmentation following Otsu binarization.

Before comparing the method with other segmentation approaches, we wanted to verify that the custom filter was transforming the original unimodal distribution of pixel intensities into a bimodal one (Figure 3.8), thus justifying the use of Otsu's binarization. We found that in contrast to the raw images, those processed with our filter displayed a more varied intensity distribution, with a small class of high intensity pixels representing the objects-of-interest being present in addition to the larger class of low intensity pixels of the background. Our filter also included pixels with intensities between the two classes which were not found in the original image, many of which were classified as interesting objects following Otsu's binarization.

Table 3.1 Mean absolute error between ground truth cell detections and detections made using each method

Method	Mean-absolute-error (MAE)	Standard-deviation
Custom filter	13.429	15.769
TrackMate	50.857	67.159
Otsu	62.429	76.778
K-means	62.214	77.734
Adaptive-G	499.143	451.845
Adaptive-M	554.214	474.238
Laplacian	385.571	240.486

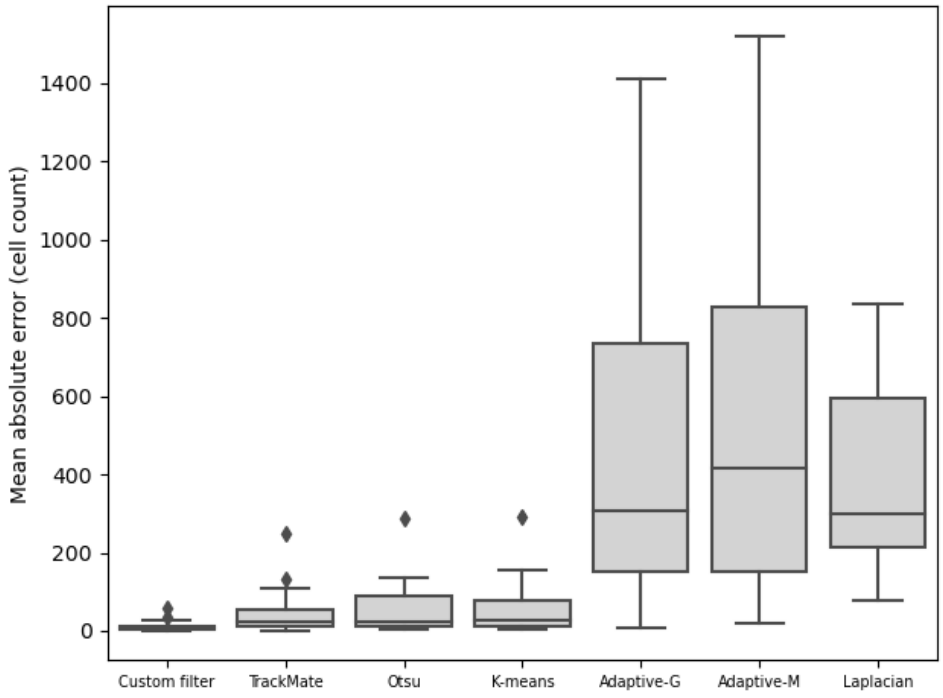


Figure 3.9 Box plot comparing the MAE of cell counts determined using of our custom filter, k-means clustering, Otsu's binarization, TrakcMate's spot detector, Laplacian of Gaussian, Local Gaussian thresholding and local mean thresholding.

We then compared the detected cell counts using different segmentation techniques with the ground truth count and spots detected using TrackMate's spot detector, as described in Chapter 2. As shown in Table 3.1 and Figure 3.9, the MAE between the ground truth and the cell count determined using each method was smallest using our custom filter at 13.429 (SD 15.769), the next highest being TrackMate's spot detector with a MAE of 50.857 (SD 67.159). This justifies the use of our custom filter against other approaches, which we chose to use as our main cell detection method moving forward.

3.4 Discussion

Our results demonstrated that by convolving images using a 9x9 sized Laplacian-style kernel with a central element value of 61, we were able in general to achieve a cell count closer to the manually counted ground truth than TrackMate for our images. Assuming that our image partitions are an accurate representation of the TLM-movies in general, we chose to use this specific approach to detect cells for cell tracking and trajectory analysis. We would not however assume that the method would be suitable for any biological context other than detecting primitive-streak derived cells in early chick development in fluorescence microscopy images, as other datasets may have different resolutions, cell distributions or SNRs to ours.

While not as accurate as our custom approach, segmentation by K-means clustering and Otsu's binarization also did not show a significant difference between the ground truth and their respective cell detection counts and were more accurate than TrackMate's spot detector. This was unexpected, as our images were shown to display a unimodal pixel intensity distribution, while both methods are designed to distinguish between two or more pixel classes and have been shown previously to perform poorly in some investigations involving cell detection [61]. It is possible that the use of a Watershed transform allowed cells to be better distinguished in our analysis, or that said investigations used images with higher noise levels than in our images.

The local thresholding approaches appeared to over-segment the images and thus the following cell detections displayed a significant difference from the ground truth, suggesting they were less suitable for cell detection than the global approaches, despite the fact the opposite is usually suggested in images with unimodal intensity distributions. The prevalence of cell fragmentation in images segmented by local approaches has been shown previously to occur, but these cells were able to be reconstituted with morphological operations [70], which is potentially an area for future investigation.

The LoG approach using a standard Laplacian filter was also significantly different from the ground truth, which alongside the interpretation of the image, was likely because a large number of regions of sharp changes in pixel intensities were being amplified, potentially resulting from noise. This contrasts with our custom approach, where the reduced central element value of the filter acted as a pseudo-threshold which required a higher intensity value to overcome the negative elements.

Since we used only partitions of a select number of our GFP TLM-movies, it is possible that in different movies or different frames, our custom filter may be sub-optimal. We however chose to use this method since the adaptive approaches we tested gave significantly different results to the ground truth and, while the model we used from Chapter 3 to estimate ideal TrackMate parameters also gave non-significant results, the significance was weaker in our custom approach as well as in Otsu's binarization and k-means clustering.

Chapter 4. Tracking groups of cardiac progenitor cells

4.1 Cell tracking background

In the previous chapter, we identified a convolution-based segmentation scheme for cell detection which gave superior accuracy results compared to TrackMate's spot detector and other commonly used techniques, forming a strong foundation for our overall tracking method. While an accurate cell detection scheme is necessary for generating accurate trajectories, linking cells across frames can be non-trivial, especially in TLM-movies with high cell population densities as shown in our datasets.

Most cell tracking methods, including TrackMate [41], use a single-cell approach, meaning they attempt to link specific cells across frames to generate a trajectory. Algorithms based on a linkage-decision must determine which cells are to be linked between frames across TLM-movies with large numbers of objects-of-interest that sometimes move sporadically. A cell may also be detected in some image frames, but missing in others, meaning the algorithm must interpolate across a gap to produce a complete track, referred to as gap-closing. Often movies involve cells that cluster closely together or overlap, so the method must also be able to handle instances where cells converge, diverge and cross-over each other's tracks.

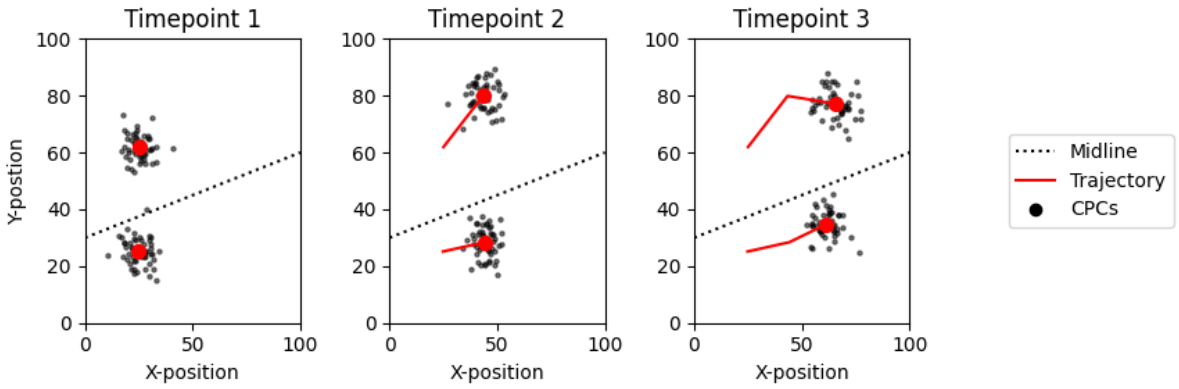


Figure 4.1 Concept of tracking CPCs as a group each side of the midline.

Rather than attempting to track the movement of each individual cell across the time-lapsed frames, we could instead track the motion between the aggregated positions of all cells within specified groups..As shown in Figure 4.1, If we first separated cells identified using the detection scheme into two groups based on whether their position is left or right of the primitive streak, we could then apply the potential tracking method to generate a generalized trajectory for CPCs each group independently to characterise the migration patterns displayed in each movie.

4.1.1 Singe-cell-tracking

There have been numerous algorithms proposed to solve many of the issues presented by tracking single cells in TLM-images. As shown in the Tracking Challenge reviews of cell tracking methods, the performance of specific algorithms in generating accurate and complete tracks can vary depending on the imaging techniques used to produce the images and factors such as the presence of fluorescent labels, image resolution, the SNR and the frame capture rate.

The suitability of a tracking algorithm may also depend on features of the cells such as their size, shape, motile activity, mitotic potential, uniformity and the distinguishability of intracellular structures. In contrast to movies in the Cell Tracking Challenge, some TLM-movies involve cells that have no visible intracellular features, do not divide and appear as fluorescent particle-like "spots". Analysing such cells could involve approaches from particle tracking, which despite being designed for fluorescent intracellular structures, attempt to solve issues regarding gap closing and overlapping events rather than lineage tracing. The tracking algorithms presented in a review of solutions to the Particle Tracking Challenge [32] showed similarities to approaches used to solve the Cell Tracking Challenge [31] and many of them are featured in popular "cell tracking" tools such as TrackMate, CellProfiler: CellTracker and SpotTracker [41–43].

An algorithm's power in generating accurate tracks depends heavily on the specificity and sensitivity of the cell detection during the preceding segmentation stage, as the tracking methods are designed with the assumption that individual cells can be continuously detected across multiple image frames. This is possibly supported by the *Ulman et al.* review [31], as for most of the Cell Tracking Challenge TLM-movies, the method with the top tracking performance included the top-performing segmentation algorithm.

Tracking algorithms generally follow different principles, with some, such as the LAP algorithm in TrackMate and CellProfiler: CellTracker following a frame-to-frame decision when linking cells, followed by a filling in of gaps [44]. The MHT algorithm in SpotTracker however follows a multiframe approach, where tracks are built across frames by finding the best fit between past and future frames [45]. Most methods are based on the Nearest Neighbour algorithm, which is usually used to identify cells in consecutive frames that are distally close, but it can also find cells that are similar based on intensity, size or shape.

In Chapter 2 we showed TrackMate's cell linkage algorithm appeared to generate inaccurate and fragmented tracks between frames, suggesting we may not be able to achieve accurate results using a single-cell-tracking method. In contrast with investigations that did successfully use such an approach for cell tracking, our time-lapsed images may have had larger SNRs, inconsistent lighting, greater cell population densities and more stochastic movement patterns.

4.1.2 Generalized-cell tracking

The proposed generalized tracking method has potential advantages over single-cell tracking, firstly being the algorithm would require less mathematical complexity and computational cost. Additionally, by tracking an aggregated position we would expect less sensitivity to occlusion events or disappearing cells, thus avoid the issue of fragmented or inaccurate tracks being generated in single-cell-trackers. The relative simplicity of the method could also allow for an easier evaluation of the accuracy of the trajectories when using annotated or simulated data, as there would be less trajectories generated to compare with a ground truth.

A disadvantage of a group-based method is that we would be unable to extract information about the migration and behaviour of individual cells, which would enable a higher resolution comparison of features between movies with different experimental interventions and the potential for larger datasets necessary for more advanced modelling using machine learning. Based on the inaccurate trajectories we generated using TrackMate however, we would not expect using such an approach to be a suitable method for extracting such information. We instead sought to focus on improving the extraction of

features previously considered by *Song et al.* [6] and preceding authors [5, 18], such as mean trajectory angles, velocity, distance and dispersion of cells for comparisons between different experimental conditions.

Except for the aforementioned studies on progenitor cell tracking, tracking aggregated positions of cells is rarely considered in most investigations. This is possibly because in most TLM-movies, cell populations are either sparse or their motion displays more stochasticity to a degree where tracking as a group would be of less value for modelling their migration [28, 54]. In the context of CPC migration however, the long-standing dogma of chick embryonic development suggests cells would be expected to form two dense clusters with a bilateral trajectory from the mid-primitive streak towards the lateral plate mesoderm [14], where a group-based approach may be more intuitive.

To give an overview of this suggested group-based tracking method as shown in Figure 1, CPC centroids identified in the cell detection stage would first be separated into bilateral groups based on their position relative to a midline representing the primitive-streak. Given that many of the TLM-movies analysed in the previous chapter contained bilateral clusters which appeared asymmetrical and the primitive streak isn't entirely stabilized across all frames, estimating a midline automatically would likely be non-trivial and require training and testing a complex model which may be too extensive given the size of our dataset. In contrast, manually defining a midline across over hundreds of frames in each movie would be too time-consuming for researchers, thus we instead considered manually defining a single midline for entire movies using the position primitive streak in the first and last frames to be more ideal in reducing computational complexity and user exhaustion.

The separation of clusters would be followed by determining an aggregated position of CPC centroids or "grand-centroid" in each frame to generate the trajectories for each bilateral group. Mean-based centroid determination is the most basic aggregation approach, however, our detection method is imperfect and cell motion can be stochastic. This means the mean-centroid position and thus the final trajectory could be heavily influenced by false discoveries, cell occlusion events and individual CPCs with non-conforming movement patterns.

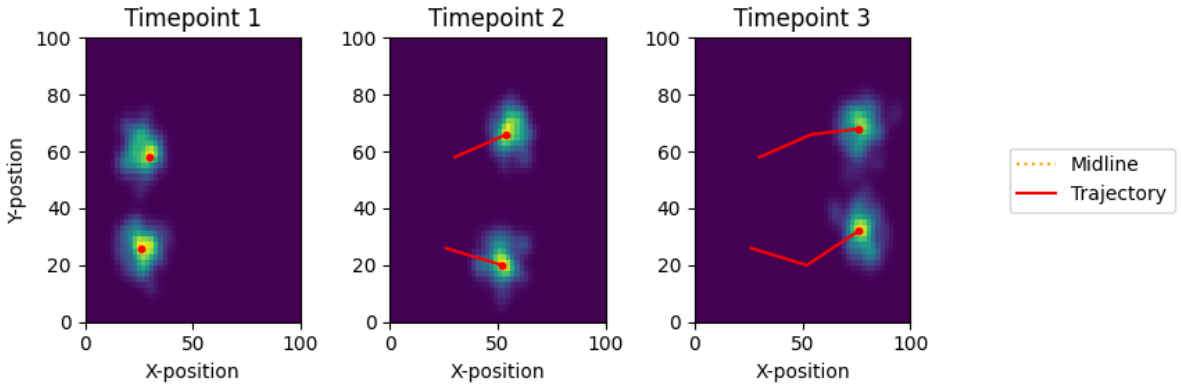


Figure 4.2 Concept of tracking cells based on the peak probability density in each frame.

An alternative approach would be to estimate the mode of the distribution of cell positions as the "grand-centroid", which we would expect to be less sensitive to noise and stochastic motion. A basic concept is shown in Figure 4.2, where the peak probability density of a group of objects can be used to track them. Mode based object-tracking has previously been performed across different contexts using mean-shift algorithms [77], which iteratively searches for the peak maxima of a non-parametric probability density function (PDF) from a specific point within a defined kernel. In this method, the peak of a PDF of a specific feature, typically pixel intensity or colour, is used to identify a starting object position in the first image frame. From this point, mean-shift is applied in the second frame to identify a new density peak within the kernel space, representing the new object position. This is repeated across each frame, generating a series of object positions that can be used to generate a track [78–80].

Mean-shift based approaches have been used previously both for cell detection [81] and single-cell-based tracking in TLMs [82] , albeit for cells which are sparser and less stochastic than in our movies. Using the PDF of all detected cell positions, it may be possible to implement a similar method to find a "grand-centroid" of CPC centroids in each bilateral cluster. Mean-shift tracking however typically involves finding the mode of pixels comprising a single-object [79], rather than multiple objects, which could lead to different requirements for our data, comprising of up to hundreds of separated single-points.

In contrast to a single distinguishable object where the peak probability density would be expected to remain relatively consistent and transition smoothly, the peak probability density of moving and occluding cell positions could be more unstable and prone to rapid transitions. While mean-shift tracking attempts to avoid these stochastic transitions using a kernel to limit the search for a peak density within a specific local window, a window size that is too small could lead to relevant information being ignored and an inaccurate trajectory being generated. It may thus be more intuitive to estimate the peak density of a distribution with no information limiting kernel windows as shown in mean-shift, or at least a kernel that is larger than traditionally used. While this would likely result in more stochastic peak maxima transitions, the overall trajectory would be more representative of the data and can still be smoothened through signal filtering [78, 80].

4.1.3 Signal filtering

While tracking the mode of a group of objects would be expected to result in a smoother trajectory than the mean, such signals are generally highly noisy even under the simplest circumstances, such as when tracking a single object in the absence of any background interference. Individual detected signals are typically considered to be inaccurate to some degree, so tracking algorithms often use signal filtering methods to reduce the noise and produce a trajectory that is less impacted by stochastic signal spikes, while still being influenced by general trends. The re-estimated object positions for specific signals are often dependant on prior signal estimates and are assumed to be closer to the objects actual position than the original signals prior to filtering.

The Kalman filter is one of the most common signal filtering methods used in computer vision when tracking objects, with it having been used in TrackMate [41], CellTrack [36], as well as previous mode-based cell tracking schemes to correct noise during tracking. The algorithm works by first initializing an original state, which in cell tracking would typically be the detected object positions and velocities in the first few time points. These are used to predict an object's state in the next time point, which is then

updated using the measured state of the object. This is performed iteratively across all time-points, generating a trajectory that is smoother and less sensitive to noisy detections.

To explore tracking CPCs as whole bilateral groups for comparing migration patterns, we chose to implement two tracking methods, the first being a time-dependant method, where cell groups would be linked frame-by-frame like in TrackMate. While this method would be ideal, we considered it possible that noisy trends across frames could generate inaccurate trajectories. We thus considered a simpler method for tracking involving superimposing frames on top of one another, then following the density across the X-axis, which can act as a pseudo-time since the movies are aligned so that migrating cells would necessarily move up the axis. This method would not allow velocity measurements but would still enable the extraction of trajectory angles and cell dispersion.

4.1.4 Simulated cardiac progenitor cell movies

An ideal training and evaluation of a cell tracking method would include ground truth datasets in which the predicted trajectories or object labels between frames are compared to those established through image annotation. In Chapter 3, we compared the ground truth cell count with the total number of generated tracks to show TrackMate was highly prone to gaps occurring within frames, which was used as an indicator that the tool was not suitable in our biological context. As our proposed method would generate an aggregated trajectory however, we could not repeat said approach and instead required a training and test dataset where the aggregate trajectory was already known to evaluate its accuracy and reliability

Manually annotating time-lapse datasets for object tracking in dense images is time-consuming and requires expertise in the given context to be able to accurately decide which objects should be linked across frames. In imaging data of our embryonic system, this would be challenging due to the high cell density, the degree of noise across frames, the potential for occlusion events and the stochastic movement of cells. While pre-annotated datasets do exist as part of the Cell Tracking Challenges [31], they are specifically focused on evaluating single-cell trackers with cells that migrate relatively independently. This is in contrast with our movies, where cells move together as clusters towards a specific location through chemotaxis.

An alternative approach to manually establishing a ground truth dataset with a known aggregate position is to simulate the data and use these to train and test our tracking method. Simulated data has been used previously in Particle Tracking Challenges [32] and is useful both for the potential to generate large quantities of data as well as being able to replicate biological systems with high cell densities. A shortcoming of using simulated data is the inability generate noise that reliably represents the stochastic movements and occlusion events displayed in our movies. For our movies however, we felt establishing a ground truth aggregated trajectory would be too time consuming and prone to error, as it would require us to manually label true CPCs as separate objects from background noise in addition to making accurate cell linkages across frames.

The previously published synthetic data available for the Particle Tracking Challenge contains objects characterised by Brownian motion, and there is little dependence between objects in movement patterns. This is in contrast with cells in our embryonic time-lapse movies where cells move in clusters, and they fail to meet the assumption of our proposed tracking method that cells migrate in bilateral

groups. For training and evaluating our method, we chose to simulate our own data and developed movies which replicated a single bilateral cluster of cells.

As our goal is to identify an aggregated trajectory representative of the movement of true CPCs within each cluster, we chose to define the aggregated trajectory first using a polynomial function. We chose a specific function that represents the arch-like trajectory of CPCs during their migration from the mid-primitive streak to the lateral-plate mesoderm. Objects representing CPCs were then added by distorting the function, where noise was added which was sampled from a normal distribution. Following this, noisy objects were added, including cells migrating in random directions and spots appearing between frames.

Using a 80:20 split for training and testing respectively, we aimed to train our method by finding optimal parameters for trajectory generation, before evaluating it on unseen test movies. We wanted to evaluate the method based on the similarity between the generated aggregate trajectory compared to that of the original polynomial function. We hypothesized that a method which converged to give a high accuracy in training and testing would be suitable for application in our true CPC migration data.

4.2 Methods

4.2.1 Simulating migrating cardiac progenitor cells

Most synthetic data for cell or particle motion time-lapses attempt to simulate stochastic objects which display Brownian motion patterns and have completely independent trajectories, as shown in the Particle Tracking Challenge datasets. These datasets are not suitable for evaluating our suggested aggregated tracking method, as it assumes that cells move towards a general direction as clusters. We thus wanted to simulate our own time-lapses with objects that follow a similar general trajectory with normal noise preventing completely identical positions.

To stimulate our data, we first defined a general trajectory our synthetic cells should follow using the following 4th degree polynomial function:

$$\boldsymbol{f(x) = 8x + 2x^2 - 3x^3 - 5x^4}$$

where x is an array representing the image frame of the time-lapse.

To simulate the cardiac progenitor cells as our objects of interest, we modified the polynomial function to create unique trajectories with positions that are normally distributed about the aggregate trajectory. The normal distribution of cell positions is critical, as our method is predicated on the assumption that the aggregate position of all cells of interest can represent the overall migration of these cells. For this purpose, we implemented a noise generator using the package NumPy, in which floating-point numbers within a defined limit were randomly sampled from a normal distribution for each frame.

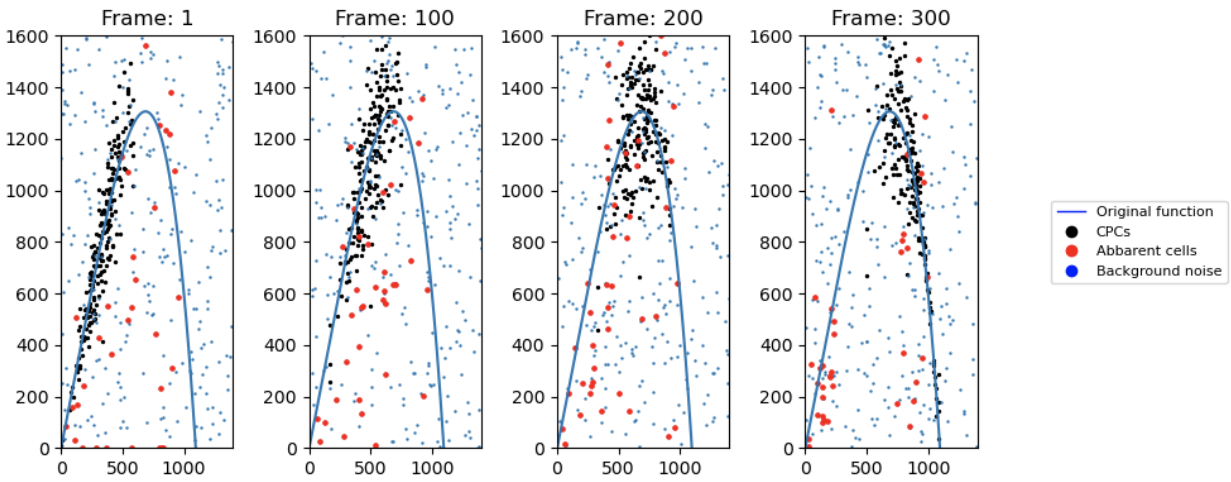


Figure 4.3 Example of simulated movies ranging frame 1-300 including the trajectory of the original function, the synthetic CPCs at positions sampled randomly from a normal distribution about the original function, aberrant cells following an "anti-function" and randomly distributed background noise.

We applied this noise generator to each of the multiples of the function, which gives a randomly distorted trajectory based on the original function. The magnitude of the distortion can be modified by increasing the limit from which the noise is sampled from, as shown in Figure 4.3, with the synthetic distortions being used to represent migrating cells. We generated 200 synthetic CPCs, in addition to 50 cells with a randomly determined function which simulates cells migrating randomly. We attempted to simulate noise by inserting 200 spots ranging 1-3 pixels in diameter at random positions for each frame.

Code for simulated movies can be accessed:

https://github.com/ConnorReynoldsUK/MScRes_thesis/blob/main/Chapter4_simulation.ipynb

4.2.2 Frametrack - tracking frame-by-frame

For testing group-based tracking, we sought to select an appropriate method for aggregating cell positions in each bilateral cluster per frame. We initially tested the least complex method using a mean-based centroid of the positions calculated using the formula:

$$\overline{\boldsymbol{X}}, \overline{\boldsymbol{Y}} = \frac{(\boldsymbol{X_1} + \cdots + \boldsymbol{X_n}), (\boldsymbol{Y_1} + \cdots + \boldsymbol{Y_n})}{\boldsymbol{n}}$$

The cartesian coordinates of the centroid $\overline{X}, \overline{Y}$ is the mean of each detected cell position both along the X- and Y-axis.

For the more complex calculation of a non-parametric multivariate PDF within each frame, we used the Gaussian kernel density estimation (KDE) function forming part of the SciPy Python statistical-based module [83]. The KDE function was initialised on the X- and Y-axis positions of the detected cell positions for each cluster, following which the estimated PDF was projected onto a 250x250 grid matrix. Each element of the resulting grid matrix represented an estimated probability density at specific cartesian coordinates. For the time-dependant tracking, we identified the point of maximum probability density in every frame and selected this as our mode-estimated aggregate position for each cluster.

The final trajectory was generated by re-estimating the aggregated positions using a Kalman filter in the Python package Pykalman [84]. We applied a double-filtering approach, where initial estimates were made using a model set to 10 iterations. These estimates were corrected was using a second model, which we optimized by grid-searching different numbers of model iterations in our simulated training movies. The model's performance was evaluated by determining the Pearson correlation coefficient (PCC) between the originally defined function and the new trajectory.

When trajectories appeared to "jump" an abnormal distance across frames, we implemented a filter to remove cells over a Euclidean distance of 100 pixels away from the previously identified point of peak density. The initial peak was set to the point of origin we determined previously.

Code for the Frametrack method can be accessed:

https://github.com/ConnorReynoldsUK/MScRes_thesis/blob/main/Chapter4_Frametrack.ipynb

4.2.3 Mudtrack – tracking using pseudo-time

To further explore the potential of a KDE approach for the logical addition of frames as used by previous authors on cell migration in early chick embryogenesis [5, 6, 18], we summed the KDE estimated densities of all frames into a single grid matrix. Our movies are all aligned so that the net movement from the primitive streak towards the lateral plate mesoderm is represented up the X-axis, meaning this can be used as a proxy for the progression of the movie. We thus used the X-axis as a pseudo-time by iterating over each column of the PDF grid and finding the peak density and the associated Y-position.

We used the peak density at each X-position, with the trajectory being generated following the optimal Kalman filter protocol described in the previous sub-chapter. We also tested using the weighted-average of the Y-positions, with the weights being the density at each X-position. To see if the model could be improved by removing lower density regions, we iteratively set a minimum threshold based on density quartile values. We evaluated the model's performance in each case using the PCC between the new trajectories and the original function.

Code for the Mudtrack method can be accessed:

https://github.com/ConnorReynoldsUK/MScRes_thesis/blob/main/Chapter4_Mudtrack.ipynb

4.2.4 Manually defining a midline in true time-lapse-microscopy movies

For our actual CPC migration time-lapses, we first aimed to establish a ground truth separation of all cells based on whether they are located left or right of the primitive streak. We achieved this by observing the bright-field images of each TLM-movie and selecting two points on the primitive streak, one of which being the most posterior point in the first frame and the other being the most anterior point in the final frame. These positions together were used to produce an estimated midline, from which we binarized cells based on their position relative to the line using a cross-product operation.

As the initial frame across movies is not fixed and many time-lapses start after HH3+ where CPCs have already departed from the mid-anterior primitive-streak, we had no uniform origin point necessary to extract exit angles from the trajectories. For this reason, we also manually selected a position located in between the most posterior point of each cell group, after which we identified a point along the midline closest to this position as an estimated origin point.

4.2.5 Unsupervised learning to separate bilateral cell groups

With the aims of achieving an automated method for separating cells based on their Euclidian distance from one another, we tested several “off-the-shelf” unsupervised learning methods provided by scikit-learn such as K-means to cluster cells into discrete groups.

Although being arguably the most simple clustering method, K-means is parametric and requires us to define a *k* number, corresponding to the number of groups our cells will be clustered into. We thus tested the method using different *k* numbers and estimated an ideal clustering result using the elbow method – a heuristic approach which iteratively tests the dispersion of clusters using different *k* number and finds the one where the dispersion curve starts to flatten, referred to as the “elbow”. Where an optimal k-value equalled 3, we sought to identify the outlying or noisy cluster by running a pairwise t-test between the x-positions of each cluster. With the assumption that the greatest variation would be shown between the noisy cluster and the bilateral clusters, we ran a pairwise t-test across all clusters. We then summed the P-values associated with each cluster before removing the cluster which had the most significant summed value.

To test how effective the automated method would be in separating CPCs into their actual bilateral groups, we compared the positions in each discrete group generated by the clustering with the two cell groups established by manually defining the midline, which acted as a ground truth. The percentage of cells labelled in the automated approaches that coincided with labels of the ground truth separation was used as an evaluation metric.

4.3 Results

4.3.1 Optimising tracking parameters using simulated data

To develop, optimize and evaluate our aggregated-cell-tracking method, we generated 100 simulated movies which contained 200 migrating cells normally distributed about a defined function, 30 aberrant cells and 200 spots representing random noise. We split this dataset randomly into 80 movies for training and 20 for testing. We first tested using our "Frametrack" method - a frame-by-frame approach where the peak density in each frame was used to represent the aggregate position. Our hypothesis was that the densest position in each frame would be an accurate way of recreating the originally defined function, even in the presence of noise.

The trajectory generated using the Frametrack approach was similar to that of the original function, giving a PCC of 0.93 (SD 0.043). As the tracks were not smooth and contained noise in the X- and Y-axis, we applied a Kalman filter with different numbers of repetitions of the second sub-model to find an optimal value in our training dataset.

Table 4.1 Performance of the Frametrack method with different Kalman repetitions compared to the trajectory generated without signal-filtering

2nd Kalman model repetition	Pearson correlation coefficient (PCC)	Standard deviation (SD)
No filter	0.9287	0.0415
0	0.9555	0.0325
5	0.9563	0.0323
10	0.9565	0.0323
15	0.9566	0.0322
20	0.9567	0.0321
25	0.9546	0.0323
30	0.9510	0.0337
35	0.9497	0.0344
40	0.9494	0.0346
45	0.9493	0.0347
50	0.9492	0.0347

Table 4.1 shows the optimal value for Kalman repetitions was found to be 20 in our training dataset, which gave a mean-PCC of 0.957 (SD 0.032). All instances of using the double-model Kalman filter gave trajectories that were closer to the original function than in the non-smoothened trajectory, the lowest being 0.949 (SD 0.034) at 45 repetitions. We therefore selected the model with 20 repetitions for a final evaluation, which gave a mean-PCC of 0.964 (SD 0.024).

We also explored our "Mudtrack" method shown in Figure 4.4, which involved summing the densities of all frames and using the X-axis as a pseudo-time. We considered this approach suitable as our movies were aligned so CPCs would be expected to progress up the X-axis during the time-lapse. We trained different Kalman models using the peak density across each X-position as well as the weighted average with different minimum density thresholds.

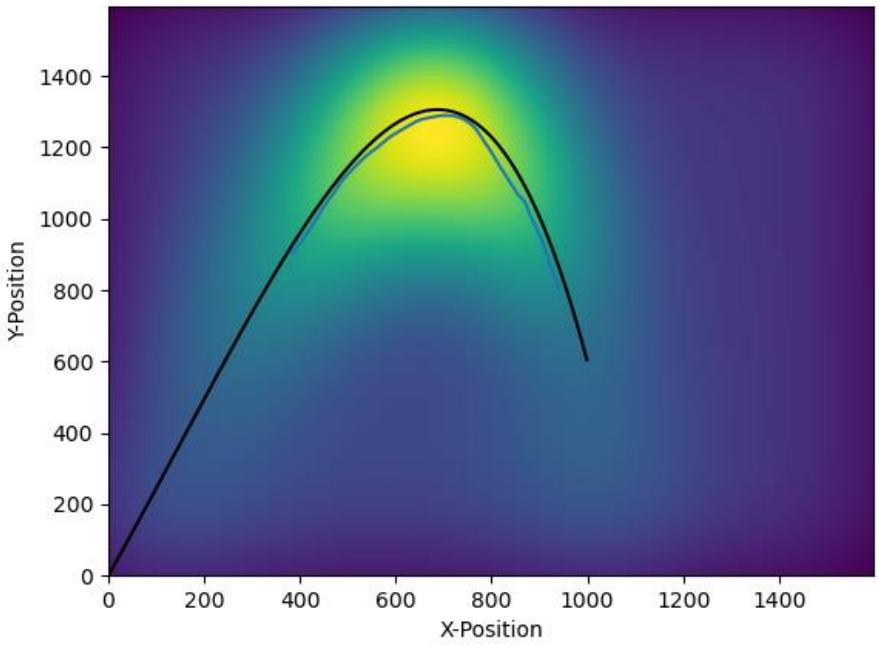


Figure 4.4 Example of kernel density estimate for cell detections using the Frametrack method in simulated data, with the original function (black) compared with the new generated trajectory (blue).

Table 4.2 Performance of the Mudtrack method using either the peak density or weighted-average as an input for the Kalman filter across different density thresholds

Density threshold (quartile)	Pearson correlation coefficient (PCC)	Standard deviation (SD)
Peak density	0.9492	0.0152
0	0.8749	0.0660
0.1	0.8633	0.0741
0.2	0.8686	0.0671
0.3	0.8805	0.0702
0.4	0.8872	0.0529
0.5	0.8774	0.0672
0.6	0.9017	0.0526
0.7	0.9048	0.0579
0.8	0.8784	0.0633
0.9	0.8440	0.0958

Table 4.2 shows that in our training dataset, the peak density approach had the largest mean-PCC at 0.95 (SD 0.02) compared with using a weighted average with all density thresholds considered. The next closest mean-PCC at 0.90 (SD 0.06) was shown using a density threshold of quartile 0.70. We thus selected using the peak density for our Mudtrack method in the final evaluation.

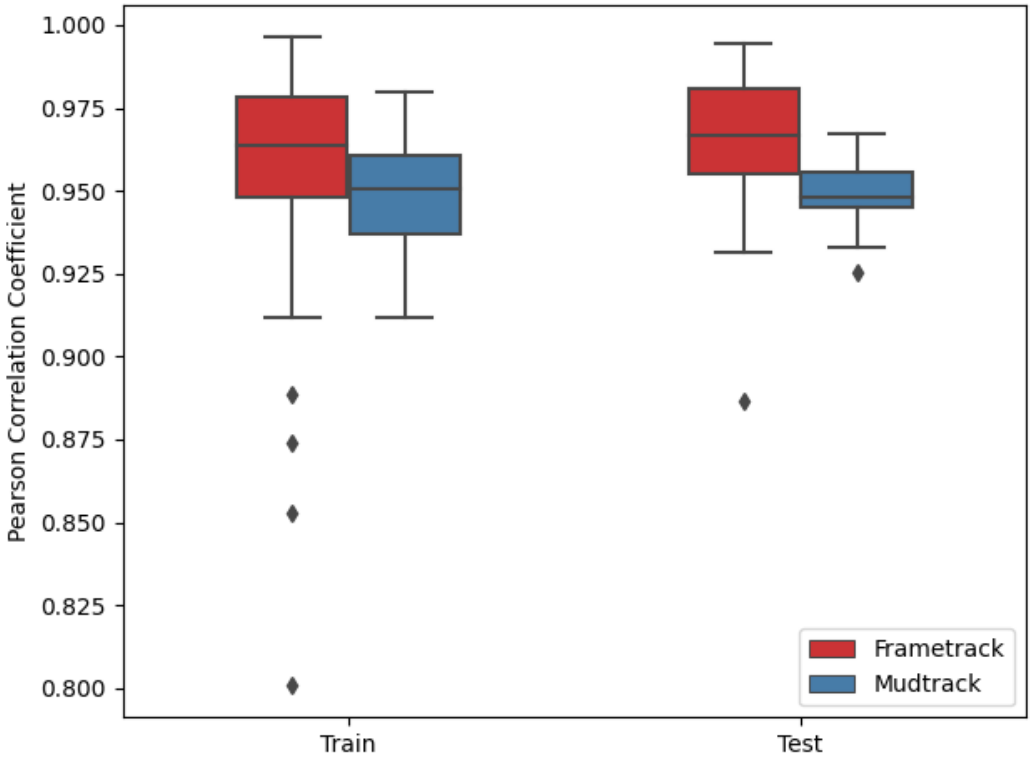


Figure 4.5 PCC values between generated trajectories and the original function in simulated training and test movies using Frametrack (red) and Mudtrack (blue).

After finding the optimal features or parameters for our model, we ran both the Frametrack and Mudtrack method using the test dataset. Figure 4.5 shows the Frametrack method was closer to the originally defined function with a mean-PCC of 0.9641 (SD 0.0251), while the Mudtrack method was less correlated with a mean-PCC of 0.9485 (SD 0.0987) in our test dataset. Although the greatest mean performance was shown in Frametrack, Mudtrack had a smaller standard deviation and less low PCC outliers in both the training and test datasets. This may suggest the Frametrack method is less robust and could perform sub-optimally in some movies, therefore we decided to use both approaches when analysing our actual TLM-movies.

4.3.2 Manually separating bilateral cell groups in actual time-lapsed-microscopy movies

In our actual TLM-movies, we applied our manual method of identifying a midline representing the primitive streak and the origin point of cells to allow a separation of CPC populations across the entire time-lapse, where we used a test set of 6 movies. We noted that while movies appeared to contain detected cells that entirely fit the migration patterns expected to be seen in early embryonic cardiac development, others include many noisy detections or cells with defective migratory behaviours.

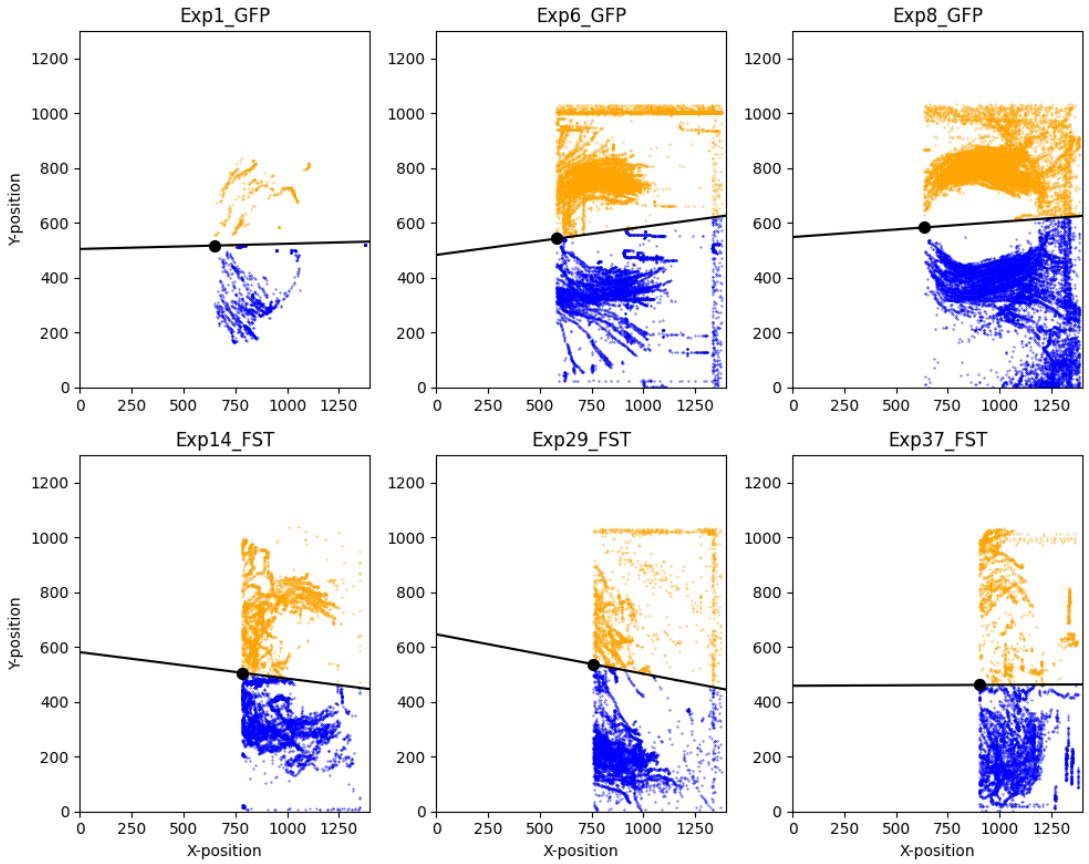


Figure 4.6 Whole-movie cardiac progenitor cell detections with positions below the origin region filtered out. Separated using a cross-product following manual definition of the primitive streak (dotted line) and origin point (black dot). Cells are separated into left (blue) and right (orange) groups.

For our analysis, we would consider the trajectory to start at the defined origin point along the primitive streak, from which cells are expected to migrate from the origin point towards the anterior of the emrbyo, moving up the X-axis. A large number of cells appeared to move from the origin point down the X-axis closer to pixel 0, which is towards the posterior of the emrbyo. Cells moving in this direction are unlikely to be contributary to cardiac tissue, as they are migrating in an opposite direction to the lateral plate mesoderm. While these abbarent cells may be of interest as a metric for defective migratory behaviour, we first sought to identify cells which were moving in the expected direction. To achieve this, all cells posterior to the defined central position were filtered out, as shown in Figure 4.6.

4.3.3 K-means for bilateral clustering

To test separating cells using automated approaches, we ran K-means clustering – generally considered the simplest unsupervised approach available, as shown applied to our movies in Figure 4.7. While we intuitively would have expected a k-value of 2 to produce bilateral groups separated by the primitive streak, we found for many movies a k-value of 3 produced a more accurate separation of CPCs based both on the visual representation as well as this being the location of the "elbow" on the dispersion curve.

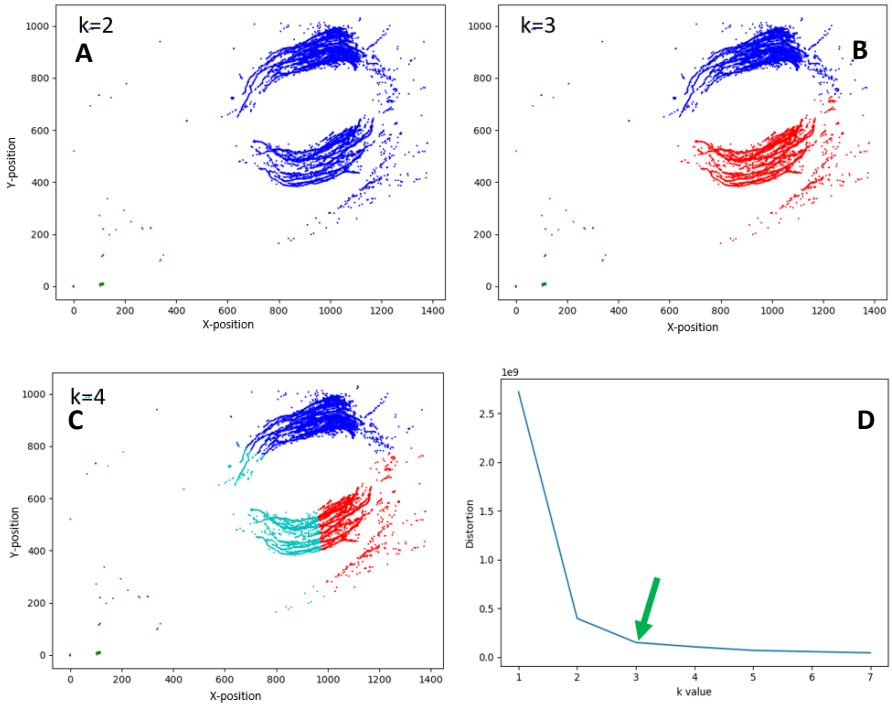


Figure 4.7 Whole-movie cardiac progenitor cell detections for Exp53_Wnt3a following clustering with a k-value 2-4 (A-C). The dispersion of clusters was calculated for each k-value (D), with the "elbow" of the curve being found as the optimal k-value of 3.

Where the optimal k-value was estimated to be 3, the additional cluster would presumably represent noise or misplaced cells. To remove these objects, we performed a pairwise comparison between each cluster and removed the one which gave the least significant summed P-value. We performed this on a preliminary test movie that had not been stabilized (and thus contained less noise), as shown in Figure 4.7 In this example, the method appeared to successfully separate CPCs into bilateral groups while removing atypical detections using a *k* value of 3.

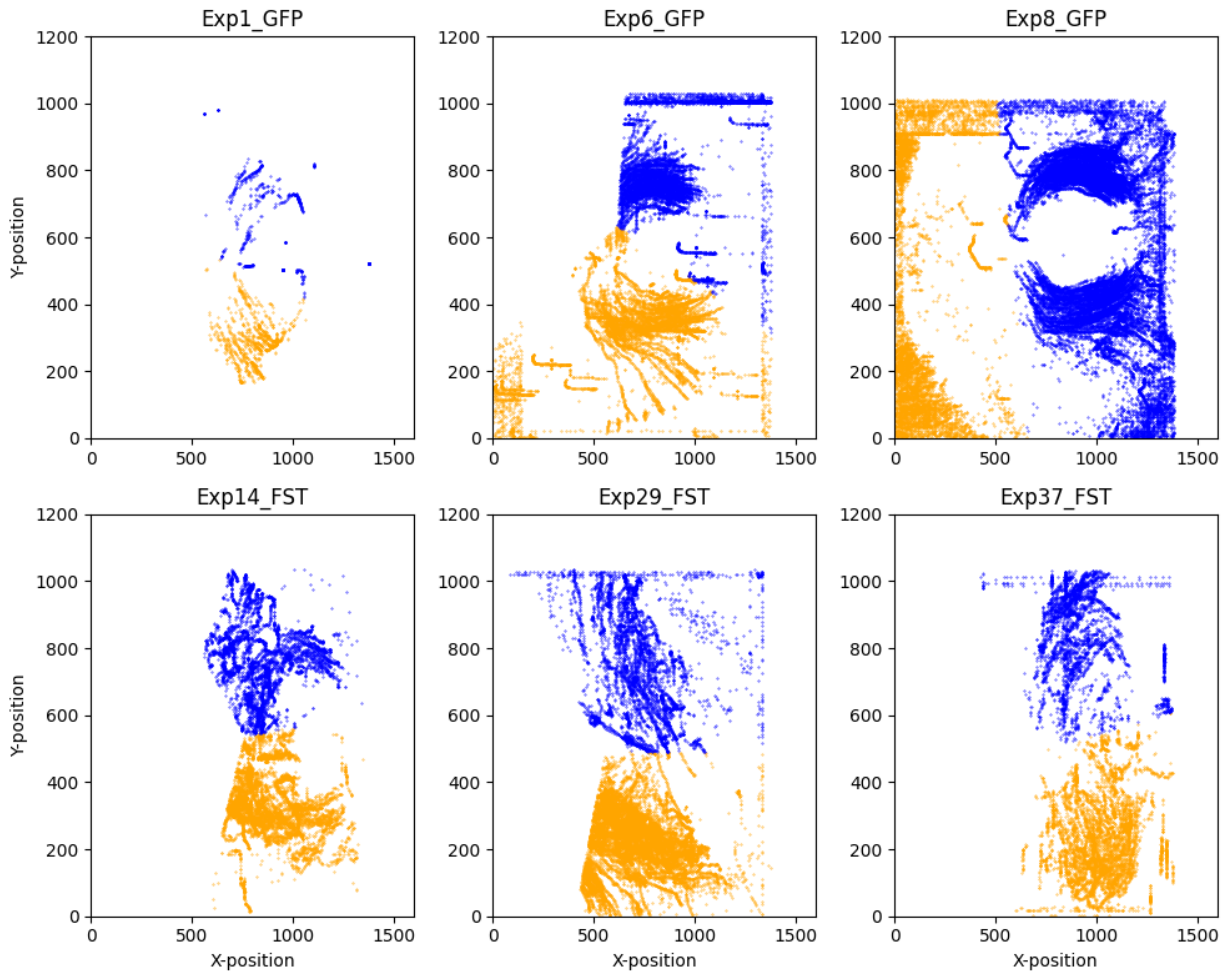


Figure 4.8 Whole-movie cardiac progenitor cell detections for test movies following clustering with a k-value 2-4.

We then ran this clustering approach on our more noise-prone stabilized training set as shown in Figure 4.8, which demonstrates obvious limitations in the method. The most dramatic flaw was shown in Exp8_GFP (Figure 4.6), where the elbow method determined an ideal *k* value of 2, yet the centroids separated the posterior-anterior regions of the embryo rather than the bilateral groups of CPCs. In all cases where the ideal *k* value was 3, the most posterior cluster of cells (closest to 0 along the X-axis) was removed. While we initially suspected this small third cluster would comprise primarily of false detections or cells with aberrant migratory behaviours, in practice the third cluster appeared to contain a large portion of cells which should constitute the CPC bilateral groups. The removal of this cluster, as most clearly shown in Exp24_GFP (Figure 4.6), could result in a large loss of relevant information for the final analysis.

To evaluate the effectiveness of the K-means separation method quantitatively, we sought to determine the accuracy of the discrete labels by comparing the total number of cells with those that were correctly labelled. The CC/GT accuracy was calculated using the unfiltered cell counts, while the CC/C accuracy was determined using cells which had been filtered following removal of the third "noisy cluster". Table 4.3 shows that with the exception of Exp8_GFP (Figure 4.6), where the clustering had failed to separate CPCs, all other movies were above 80% accuracy and the difference between accuracy metrics did not exceed 10%.

Table 4.3 Evaluation of K-means clustering comparing the accuracy of labelling using all detected cells and those which had been filtered by removal of the third cluster.

Movie	Ground truth cell count (GT)	Cluster cell count (C)	Correctly clustered cell count (CC)	CC/GT Accuracy (%)	CC/C accuracy (%)
Exp1_GFP	1374	1258	1104	80.35	87.76
Exp6_GFP	11868	11013	10523	88.67	95.55
Exp8_GFP	19375	18716	9905	51.12	52.92
Exp14_FST	8846	8527	8218	92.90	96.38
Exp29_FST	6568	6264	6163	93.83	98.39
Exp37_FST	4946	4796	4492	90.82	93.66

While K-means can be considered successful in separating CPCs into bilateral groups for most movies, the cells lost by the removal of the "noisy cluster" could result in major differences in the tracks generated compared to when using the cross-product labelled cells. Additionally, defining a midline automatically is non-trivial and would likely require near perfect accuracy results greater than 95% to achieve a line similar to the manually selected one. As the midline is essential for calculating the angles and width of trajectories during cell migration, we discarded the K-means approach and instead focused on using the manual approach with cross-product separation as described in 5.4.2.

4.3.4 Generalized-cell tracking

While we are unable to quantitatively evaluate applying the tracking methods previously described on our movies, we are able to visualize the trajectories generated to see if they meet the assumptions of CPC behaviour. We expected to see trajectories that move smoothly from the point of origin on the mid-primitive streak up the X-axis pixels towards the lateral plate mesoderm.

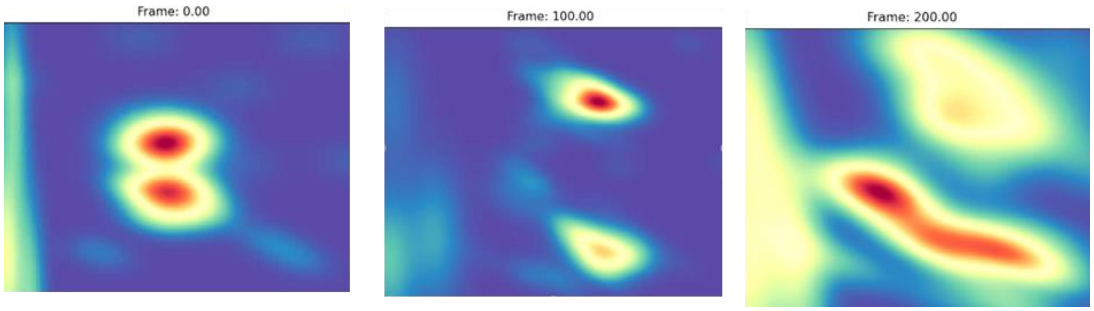
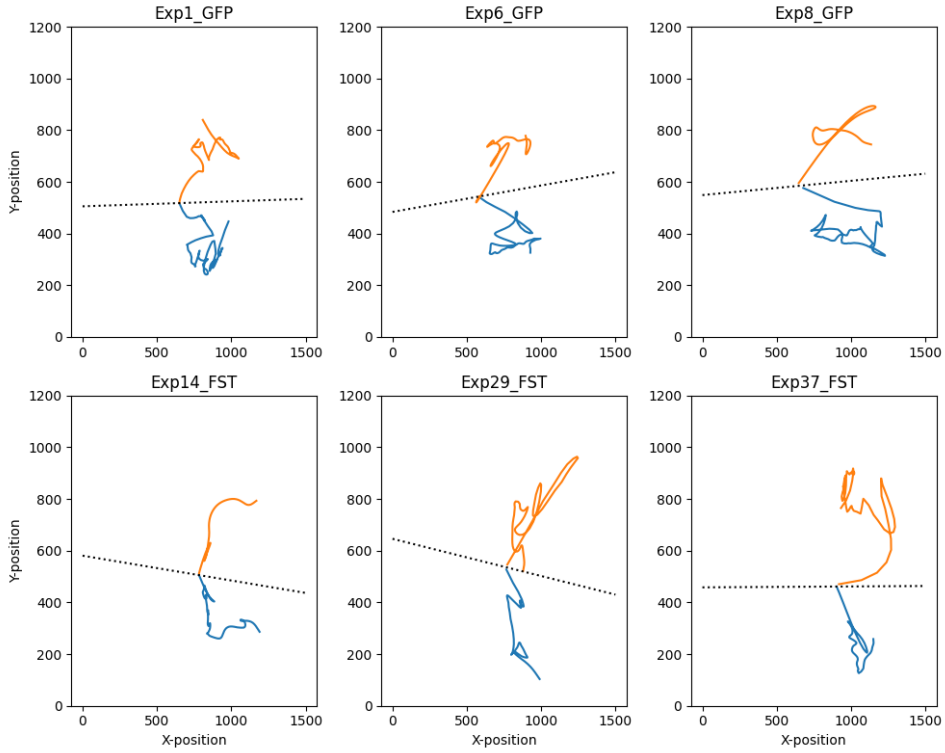


Figure 4.9 Frametrack approach showing KDE estimates for cell detections across 200 frames in Exp8_GFP.

We first applied the Frametrack method on our real TLM-movies using the same parameters as described for the simulated movies (Figure 4.9). When observing the density estimates of detected cells across frames we noticed that in later frames, the density at higher X-positions where the lateral-plate-mesoderm is located diminished. This is possibly because the GFP expression is transient and cells which complete their migration may have reduced expression.

A



B

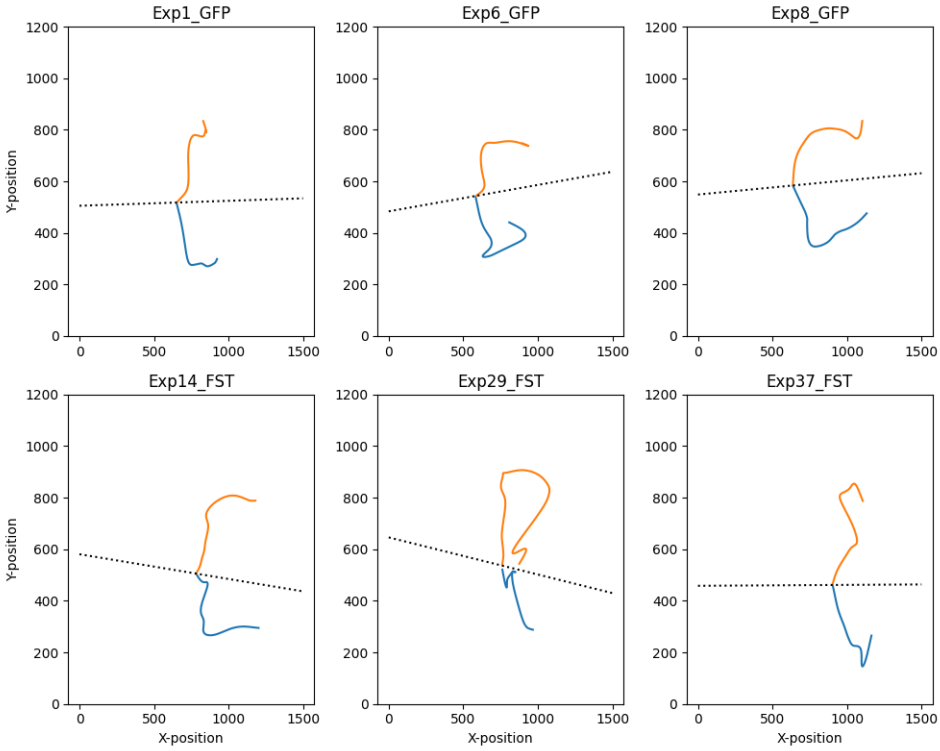


Figure 4.10 Frametrack approach using all cell positions (A) as well as using only cell positions 100 pixels from the originally defined peak density (B).

In contrast to what was expected, the Frametrack method appeared to generate trajectories which "jumped" long distances between frames (Figure 4.10A). In all cases except for Exp14_FST, the aggregate position tended to move down the X-axis in reverse towards position 0 at some point, which is the opposite direction normal CPCs should move in. This suggests these true movies contains more stochasticity than was accounted for in our synthetic datasets.

To correct the issue of long-range "jumping" between frames, we applied a bounding region around each previous position, as inspired by mean-shift approaches (Figure 4.10B). Detections that lay outside this region were filtered out, which we assumed would lead to trajectories that moved in a consistent direction. This change gave us trajectories that moved generally in the expected direction towards the lateral-plate-mesoderm, this movement didn't capture the full long-range migration of cells and differences in CPC migration across GFP-control and FST movies were not obvious visually.

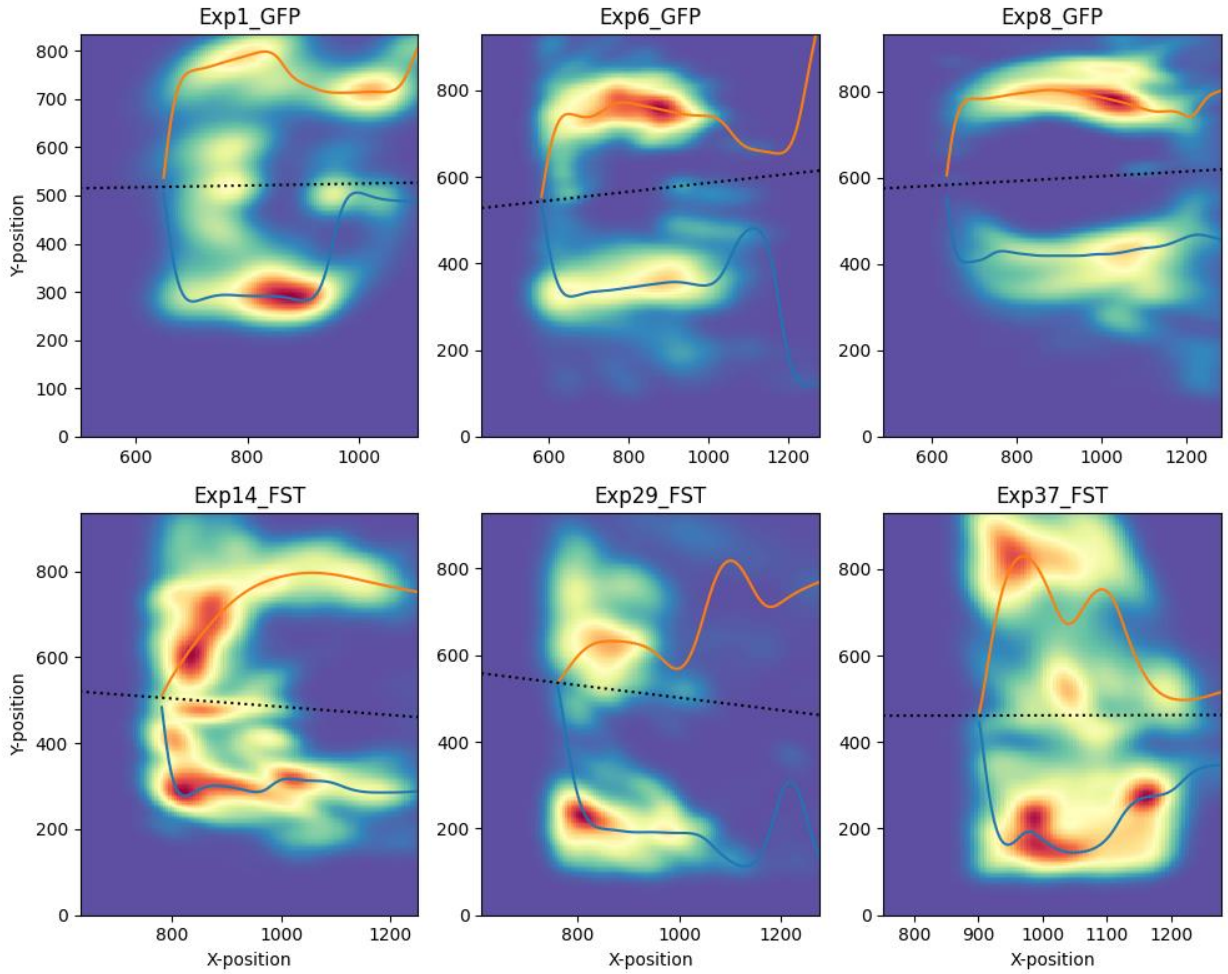


Figure 4.11 Mudtrack approach using parameters optimized on synthetic datasets on our true CPC migration movies.

We also applied the optimized Mudtrack method as shown in Figure 4.9, which was able to capture a more complete migration trajectory across the time-lapse than what was observed using the Frametrack approach, though noise at higher X-axis positions did appear to disrupt the trajectory, as shown in Exp1_GFP and Exp37_FST (Figure 4.11). As shown in all movies, the trajectories become stochastic in regions where no cells are present, as the density is too low for Mudtrack to converge. It appeared that there was a "cut-off" point at X-axis after-which the trajectory is no longer a valid representation of the cell's movement

Visually, we could not distinguish differences in trajectory between GFP or FST other than the former appeared more symmetrical, however this is likely more an issue with defining the midline rather than observing a phenotypic difference.

4.4 Discussion

We were able to produce two methods involving generalized-based tracking which were optimized and evaluated using synthetic datasets. The simulated time-lapsed images were inspired by the visual characteristics and behaviours of migrating CPCs demonstrated in previous literature, but our synthetic datasets likely lack some of the more disruptive noisy features present in the actual CPC migration movies. This was demonstrated by the fact the Frametrack method performed well in the synthetic movies but poorly in the true movies. Developing a method to evaluate how representative these synthetic movies were of true CPC migration movies would be non-trivial, but future simulated data could attempt to manually annotate cells and noise in a select few movies to better model our true datasets.

Applying these methods on the test data suggested there were some problems with the lighting quality and stability of our movies. In both Frametrack and Mudtrack, the trajectory became more stochastic in later frames. For future analyses of our movies, we therefore chose to use only the initial trajectory. We also decided to apply the Mudtrack method rather than Frametrack, as it visually appeared less sensitive to the noise which we were not able to account for when simulating the synthetic movies. This is arguably less ideal than the Frametrack method, as the Mudtrack method cannot determine the aggregated velocity of the cells. We can still however extract other features such as trajectory width, dispersion, and exit-angle which we believed to be more accurate than in the noise prone Frametrack method.

The use of a generalised approach could also be argued to be less ideal than a single-particle-tracking approach, since in the latter we would be able to characterize the behaviours of individual cells which would allow for a more detailed phenotype to compare across different experimental interventions. Ultimately however, we felt that since single-cell-trackers cannot accurately generate complete tracks for individual cells in our movies, they would not be suitable either for investigating individual cell behaviours or generalized cell trajectories, while our generalized approach is able to least characterize the latter.

5. Comparing cardiac progenitor cell migration patterns

5.1 Comparison background

Having defined our Mudtrack method for generating trajectories for migrating CPCs, we next sought to extract meaningful features from said tracks to make comparisons between movies with different experimental interventions. In our movies, embryos are either labelled “GFP”, representing controls with no treatments, in addition to those with overexpression treatments for morphogenic signals including “FST”, “Wnt3a” and “BMP2”.

Previous work by *Song et al.* [6] included comparisons using both the CPC’s distance from the midline and “exit angle”. While it would be simple to extract a pixel-level max distance from the mid-primitive streak for each trajectory, converting this into an actual distance to allow for comparisons is problematic as for many of the movies the resolution is not specified. In the absence of this information, we considered using distances in movies that may have different resolutions to be unsuitable as features for comparisons.

As we have a defined CPC origin point along the midline, we would be able to determine the angle of the aggregate position of cells along any point of the trajectory. Using angles as a feature has advantages over distance, since it does not have to be corrected to account for differences in resolution. The “exit angle” is described as being determined for the initial 10 coordinates along the CPC trajectories, but we felt this would be too arbitrary of a selection for our movies, as some have different starting points. We instead considered using all angles from the origin point to the first maxima distance from the midline of a trajectory to compare across movies with different starting stages.

To ensure the application of a method which tracked cells as aggregated groups was justified, we wanted to again test TrackMate’s single-cell linkage method specifically whilst disregarding its cell detection approach. TrackMate has previously been applied following a pre-segmentation step in contexts to avoid shortcomings in the LoG spot detector [54]. Hence we wanted to test how processing our movies using the custom convolutional filter described in Chapter 3 would affect the trajectories generated, which we previously established in Chapter 2 were highly fragmented to point of being unusable.

A direct comparison between Mudtrack and TrackMate is not possible, considering the fundamental differences between tracking cells as groups and tracking cells as individuals. In the former, we generate two continuous trajectories across the time-lapse while in the latter, most tracks are expected to have gaps between frames. Regardless, we can still quantify how much coverage different tracks have across frames and visually decide if any of the more continuous tracks present a valid alternative to Mudtrack’s trajectories.

The low number of samples has meant we focused on implementing mechanistic methods, rather than using more naive supervised machine learning models in our comparisons. With fewer than 50 usable movies, training a reliable predictive model would be unlikely even using the most basic regression models. As more TLM-movies of migrating CPC are generated in the future however, machine learning approaches could be a highly useful method for automatically extracting phenotypic information from the images.

We wanted to develop a basic pre-processing method to prepare our movies for input across different machine learning classifiers as a blueprint for future work. The classification problem was chosen to be a simple binary between GFP-controls and BMP2/FST-overexpressed embryos. This basic classification was decided since we do not have any continuous ground-truth phenotypic labels to use. Regardless, the output of a binary classifier is based on a probability that would be expected to be related to a phenotype (assuming the model is not biased towards noisy features in the image). The binary classification is therefore a good initial first step to see if the models are capable of learning information from our movies.

5.2 Methods

5.2.1 Extracting exit angles from Mudtrack trajectories

To determine the initial exit angles, we used the trajectories generated for CPCs in only high-quality time-lapses as described in Chapter 4. As the midline (representing the primitive-streak) is not entirely horizontal along the X-axis, we first had to ensure each angle measured along the trajectory intersected the midline at a 90^{o} angle as shown in Figure 5.1. This step is necessary as if the midline is diagonal one bilateral trajectory will be more acute than the other, thus preventing a valid comparison.

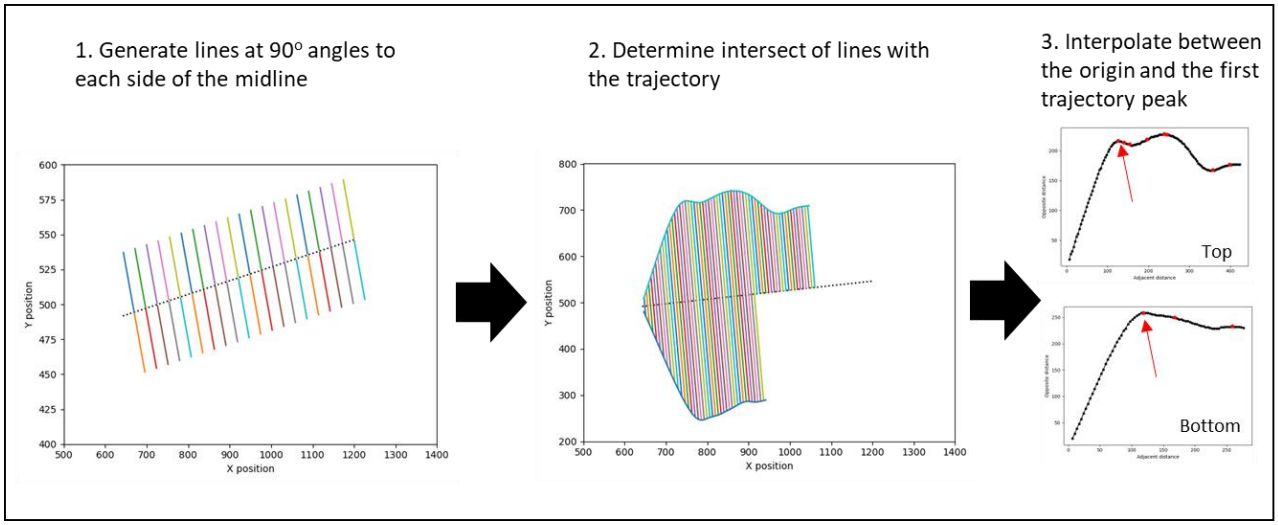


Figure 5.1 Method for extracting exit angles between the origin point and the initial peak maxima of the trajectory

We extrapolated positions at a 90^{o} angle from the midline and found the point at which the trajectory curve intersects. We then calculated the angle of each position adjacent to the midline using an inverse cosine function. To determine which angles we would use as features, we identified the first peak maxima of the distance of each position from the midline and used angles only found between the point of origin and this initial peak. We also interpolated the angles to an array of length 100 to ensure features were comparable across different trajectories.

Based on the results of previous literature [6], we assumed that the exit angles should be symmetrical and that any instances of asymmetry were a result of errors in the definition of the midline. We therefore averaged the exit angles found for each group of bilateral CPCs. Significance tests were made between each position using Student's T-test, following which P-values were combined for each movie using Fisher's method.

5.2.2 Pre-segmenting movies for input in TrackMate

We initially wanted to compare TrackMate trajectories directly to those generated in our Mudtrack method. For a fair comparison we needed to ensure the image processing techniques used were the same, as different processing techniques would be expected to lead to an unequal cell detection accuracy. This can greatly alter the accuracy and coverage of tracks across frames, so we took the same movies used in 5.2.1 and pre-segmented them using our custom convolutional filter described in Chapter 4. In the absence of any pre-established ground truth trajectories, we instead chose to analyse how much coverage each track had across frames. This was determined by finding the percentage of frames each track was present in across the whole time-lapse. We assume that coverages under 75% would present challenges to extract any information from considering a complete analysis would require tracks to be aligned by time-points.

5.2.3 Development of a machine learning classification framework

We wanted to test multiple classification methods using different supervised machine learning models to see if a framework could be established for future work. The classification task was a simple binary problem of predicting whether a movie was a control or had an experimental perturbation. We first divided our movies into a 3:1 training and test split to ensure the final evaluation of each model was unbiased.

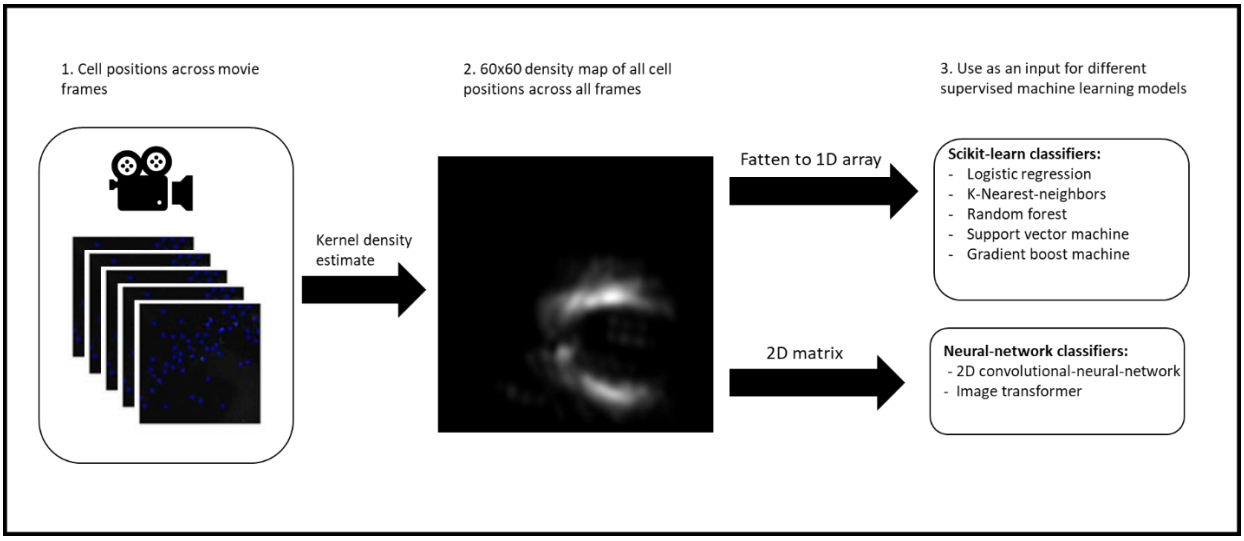


Figure 5.2 Framework for processing microscopy movies by combining cardiac progenitor cell positions across all frames before conversion into a 60x60 density map for input in different machine learning models

Our raw movies exceeded 1000x1000 pixels and contained up to 300 frames, which only a few types of architectures such as 3D convolutional-neural-networks (CNN) would be able to train on. This architecture is however highly complex and would be computationally costly. To avoid these costs and make the input applicable to more basic model type, we first found the centroid position of all cells across all frames in each movie using the cell detection scheme described in Chapter 4. We then used KDE with a bandwidth of 0.05 to generate a 2D 60x60 density map of cell positions in each movie (Figure 5.2).

We trained a collection of classification models from scikit-learn [49], including logistic regression, K-nearest-neighbour, random-forest decision trees, support vector machine and a gradient boosting machine. For these models, the 2D density maps were flattened so a 1D array was used as an input. We used grid-searching with a leave-one-out cross-validation approach to optimize model hyperparameters based on the F1 score. We used F1 score as a harmonic mean between classes rather than just the accuracy of the whole test dataset to account for an imbalance said classes.

For more complex machine learning architectures, we implemented neural network models using Tensorflow [85]. We first developed a 2D-CNN with two 2D convolutional layers followed by a dropout, pooling, and feedforward layer. The 2D-CNN model contained 20,108,738 total parameters, while hyperparameters were optimized heuristically using cross-validation F1 score, including a learning rate of 0.001 and a batch size of 1. Each convolutional layer had a kernel size of 3x3 with 64 filters.

We also wanted to apply a state-of-the-art vision transformer model as described by *Dosovitskiy et al.* [86], which has the advantage of being computationally efficient by dividing the image into patches while also being able to make relationships between distance features in each patch. This is opposed to 2D-CNNs, which can only learn using local feature relationships. We implemented the vision transformer mostly described by the Keras Team's Github image classifier example [87]. Our final model had a learning rate of 0.001, a batch size of 3 and contained 9 image patches each sized 16x16. During testing, embeddings were obtained from the output of the 2nd from last layer while attention scores were extracted from the final multi-head attention layer for model interpretation.

5.3 Results

5.3.1 Comparison of Mudtrack exit-angles across experimental conditions

CPC exit angles tended to be highly uneven using our method, which is in contrast to the study by *Song et al*, where the left- and right-bilateral trajectories were relatively (though not perfectly) symmetrical in non-site-specific morphogen overexpressed embryos and our movies were expected to display said symmetry. As the morphogenic overexpression was not localized to one side of the embryo in our movies, we averaged the angles on the assumption that the CPCs do migrate symmetrically and that the midline was not accurately established.

We stratified the symmetrical angles for all movies via PCA as shown in Figure 5.3B, which demonstrated most GFP-controls and those treated with FST/BMP2 clustered independently along PC1. We additionally generated a distance matrix based on the absolute difference between the mean angle of each movie. As with the PCA, hierarchical clustering also showed control-GFP movies tended to separate from movies

with embryos with FST or BMP2 treatment except for Exp22_GFP and Exp44_FST (Figure 5.3A), which were outliers to this rule.

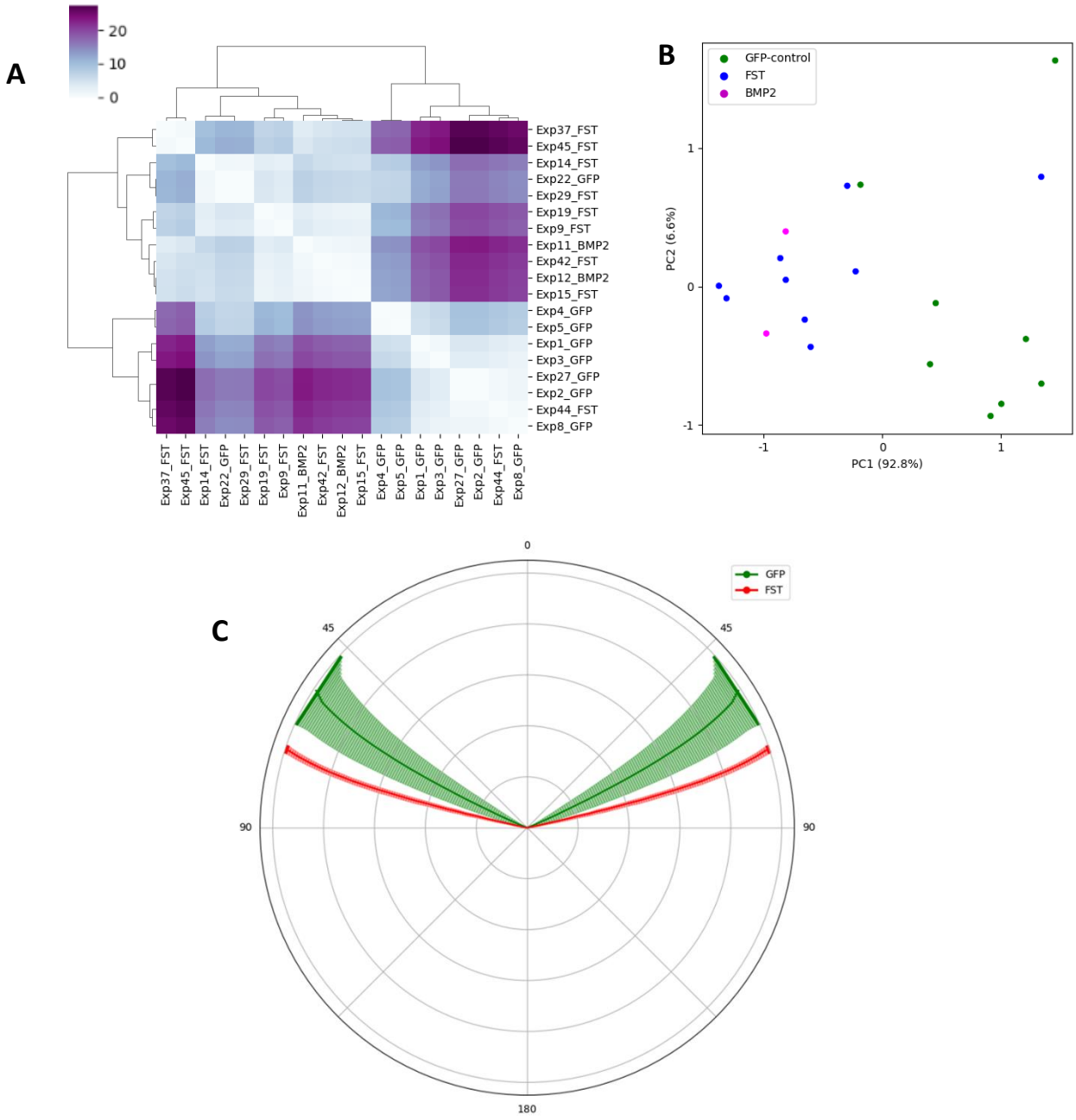


Figure 5.3 Hierarchical clustering of differences between symmetrical angles (A) and PCA plot of symmetrical exit angles (B) across movies. Polar plot showing symmetrical exit angles determined in GFP-control (green) and follistatin overexpressed embryos (red) (C).

As shown in Figure 5.3C, CPC trajectories in the GFP control time-lapses displayed a more acute grand-mean exit angle at 60.72° (SD 5.21°) compared with those treated with FST, which gave a grand-mean exit angle of 74.29° (SD 7.37°). This difference was found to be significant according to Fisher's method for combined P-values ($P > 0.0001$). Embryos treated with BMP2 had a grand-mean exit angle of 77.82° (0.62°), but only 2 BMP2 movies were successfully analysed, thus preventing statistical significance testing.

5.3.2 Evaluating TrackMate trajectory's generated using pre-segmented movies

We next wanted to rerun TrackMate's linkage algorithm on movies which had been segmented using the convolutional filtering steps described in Chapter 3 and test if these improved results shown in Chapter 2. This final evaluation was made to see if our alternative method of tracking groups of cells was indeed justified rather than the standard single-cell tracking approach. It was shown previously shown TrackMate had a tendency to generate tracks which were fragmented, so we again used the difference between maximum detected cell counts in any given movie frame with the total number of trajectories as an error proxy.

Table 5.1 Coverage of object tracks across frames

Movie	Frames	Frame coverage			
		> 25%	> 50%	> 75%	100%
Exp1_GFP	205	7	7	2	1
Exp2_GFP	205	3	3	0	0
Exp3_GFP	205	12	12	1	0
Exp4_GFP	205	7	7	0	0
Exp5_GFP	224	1	1	0	0
Exp8_GFP	190	5	5	0	0
Exp9_FST	321	3	3	1	0
Exp11_BMP2	280	1	1	0	0
Exp12_BMP2	199	0	0	0	0
Exp14_FST	297	1	1	0	0
Exp15_FST	327	1	1	0	0
Exp19_FST	105	8	8	3	0
Exp22_GFP	172	468	468	121	3
Exp27_GFP	149	653	653	64	1
Exp29_FST	249	3	3	0	0
Exp37_FST	170	3	3	1	0
Exp42_FST	204	5	5	2	0
Exp44_FST	199	3	3	0	0
Exp45_FST	99	81	81	13	0

We wanted to check how many trajectories were unfragmented, or at least find an appropriate set of tracks which covered more than 75% of frames across each movie. Tracks with high frame coverage could potentially be used as representatives to calculate an exit angle as was shown in 5.3.1, that is assuming they consistently linked cells from early frames and displayed a bone fide migratory pattern. Our results in Table 5.1 however show that in most movies, only a handful contained any trajectories covered more than 75% of the full time-lapse. The few trajectories which did contain over 75% coverage were either almost entirely stationary as shown in Exp1_GFP, or displayed stochastic migratory patterns as shown in Exp22_GFP, Exp27_GFP and Exp45_FST. Of these, TrackMate appeared to be highly sensitive to noise in both Exp22_GFP and Exp27_GFP, as these movies contained trajectories which did not appear to reflect the expected migratory behaviour of CPCs.

While tracks with > 50% or 25% coverage were more abundant, the lack of alignment across frames would leave a very small window to extract information from. Based on this, and the fact many tracks still appeared to display stochastic patterns, we considered our density based aggregate approach to be more suitable for analysing CPC migration in these movies.

5.3.3 Evaluation of machine learning classifiers for comparing movies

We finally wanted to test applying different supervised machine learning approaches to our movies to see if they could learn any meaningful information. As this purely a proof-of-concept rather than a replacement for our proposed tracking method, we started with a classification problem of predicting whether a movie is GFP or contains FST/BMP-overexpression. We generated a 60x60 density map for each movie based on the cell positions across all frames. We then split the movies into 17 training and 9 test samples.

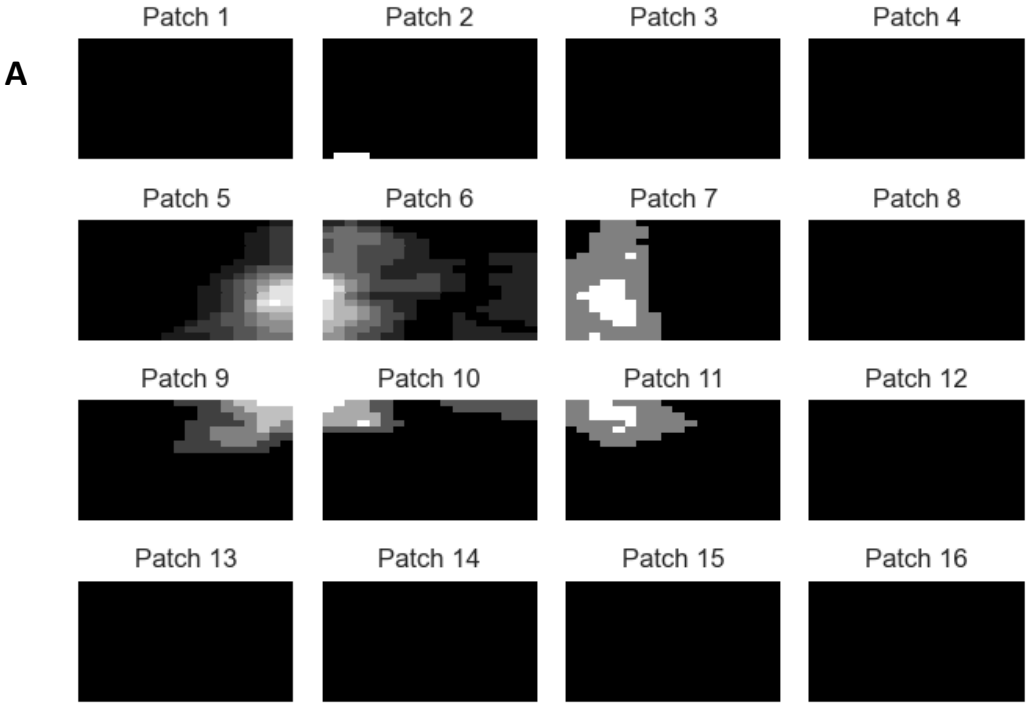
We first trained basic binary classifier models from scikit-learn including logistic regression, k-nearest-neighbour, random-forest, support vector and gradient boosting machines. We optimized the hyperparameters based on the cross-validation accuracy of each model using a grid search. Using flattened 1D-arrays of our density maps, we were able to achieve F1 scores of up to 0.88 in random-forest and gradient boosted decision trees as shown in Table 5.2. We also applied a relatively basic neural-network model in TensorFlow which included two 2D convolutional layers followed by a single feed-forward layer. The model was again optimized through cross-validation and gave a test F1 score of 0.86.

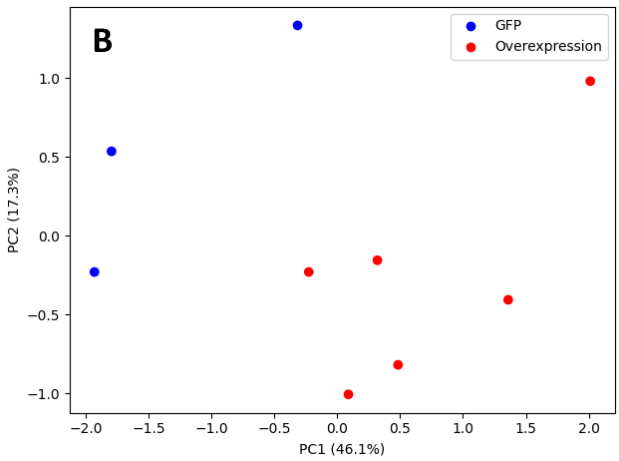
Table 5.2 F1 scores of different machine learning classifiers

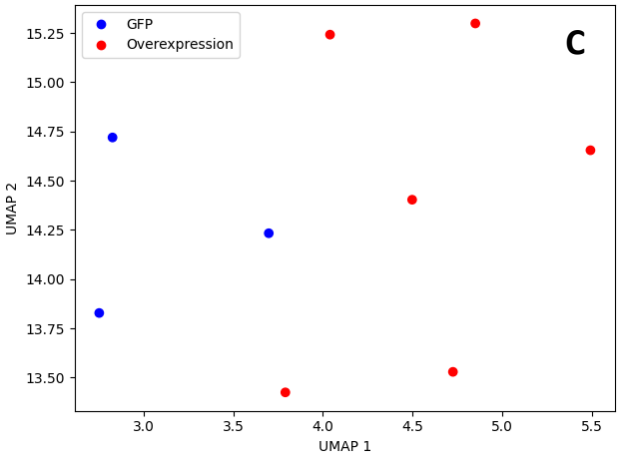
Model	F1 Score
Logistic regression	0.86
K-nearest-neighbor	0.64
Random forest	0.88
Support vector machine	0.86
Gradient boost machine	0.88
2D convolutional-neural-network	0.86
Vision transformer	**0.92**

We then used a more advanced model in the form of a vision transformer. The model generates 9 16x16 patches from each density map, as shown in Figure 5.3A, before being encoded and passed through a transformer layer. The final model achieves an F1 score of 0.92, which was higher than any of the other trained classifiers, though this is likely by pure chance as the model is far more complex than would be typically used for such a small dataset.

The vision transformer architecture also allows us to interpret the results to some degree, as the embeddings for each input are directly related to the binary output but provide continuous data which could allow us to potentially quantify a certain phenotype rather than just classify it. We applied different dimensionality reduction techniques to this method as shown in Figure 5.3C-D. In PCA and UMAP, the test samples separated correctly between GFP-controls and the overexpression embryos. Using TSNE, a single GFP movie was mixed with the overexpression samples, which was also the sample that was misclassified by the model.







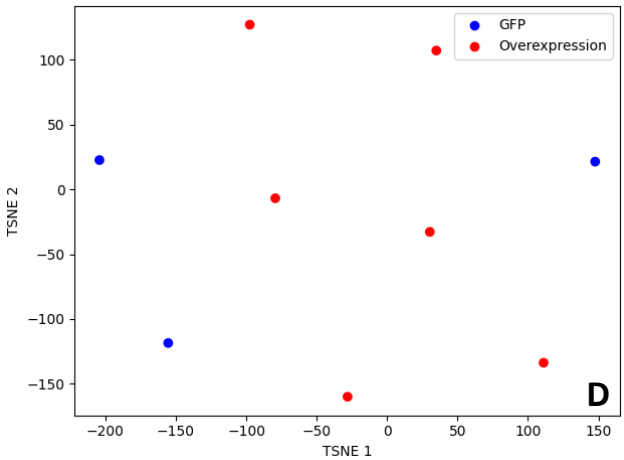


Figure 5.2 Density map cut into 9 16x16 patches before being encoded by the vision transformer (A). The decomposed vision transformer model embeddings of each test sample using PCA (B), UMAP (C) and TSNE (D).

As the vision transformer model contains a multi-head-attention block, it is possible to determine which regions of each density map gain the most "attention" from the model, which means more contributary to the prediction. This is useful for ensuring the model is learning from the correct information in our images and not any noisy features. Figure S5 shows the attention scores for an example image across the different patches, although it was difficult to interpret what features the model was attending to most. We would expect that with a larger sample size, the attention map could be more interpretable.

5.4 Discussion

We were able to apply our Mudtrack method to show follistatin overexpression significantly alters the trajectory of CPCs in general and displays a more obtuse exit angle. It has been shown previously that follistatin treatment to chick embryos prior to HH5 impairs cardiogenesis [21], though specific differences in CPC migratory behaviours had not been previously characterized before. While we were not able to establish statistical significance, BMP2 overexpression resulted in CPC exit angles close to those displayed by follistatin treatment, which coincides with the findings of the previous study by *Song et al.* The reason for similarities in phenotype between follistatin and BMP2 is possibly because follistatin is an antagonist of activin, which is known to be a critical morphemic signal in cardiogensis and establishing mesodermal fated cells in general, and is itself an antagonist of BMP2 [20].

In contrast with the *Song et al.* paper [6], our method gave more obtuse exit angles in general by around 30^{o} in the GFP-control embryos. This is possibly due to the fact all movies in said paper started prior to CPCs exiting the mid-primitive streak, while in our movies most CPCs had already exited it and began their migration, meaning we are missing the very initial exit. We considered that as the exit angles are relatively consistent within each morphogenic group and as the magnitude of the differences between interventions is comparable to previous literature, the method is still appropriate for characterising migratory behaviour. Additionally, our approach was evaluated using synthetic test data, while the methods described by *Song et al.* were not detailed and had not been benchmarked in said paper.

While the Frametrack method outperformed the Mudtrack approach in recreating accurate aggregated trajectories in our synthetic cell migration datasets, when we applied it to our actual time-lapses the trajectories generated appeared highly prone to high densities occurring in random positions as a result of noise and lighting issues. As techniques in fluorescent time-lapse-microscopy continues to advance, the Frametrack method could be evaluated in future movies. Alternatively, we could develop more advanced methods for generating synthetic datasets which replicates the noise displayed in our movies to further optimize and evaluate a mean-shift inspired tracking approach.

The machine learning results are promising as an example of how to process our movies for input across different models, but the high F1 scores alone aren't enough to suggest our trained models would reliably test different hypotheses associating CPC migratory patterns with different experimental conditions. We maintain an assumption that our dataset is too small to fully evaluate any specific model considering there are only 9 test samples. The models may simply be learning from unknown biases in our images, such as different lighting conditions between GFP-controls and FST/BMP-overexpressed embryos. We instead consider this a framework for future analyses.

6. Future work and conclusion

6.1 Image quality auditing

For testing hypotheses across different time-lapsed images, it is advantageous when the movies being analysed have high resolution, good stability and consistent lighting or background noise. Many of the movies available for our investigation had poor resolution, contained embryonic structures that moved in some cases more abruptly than the CPCs and contained multiple different lighting conditions that often fluctuated within even a single movie.

In this project, we discarded movies relatively heuristically based on features that were likely to distort our results, such as having less than 10 cell detections, displaying cells that move in the opposite direction to the lateral-plate-mesoderm or possessing high-intensity artefacts across frames. While this is achievable for datasets of our scale, this could be highly time-consuming as more time-lapsed images are generated. For high-throughput analyses, future researchers could consider developing machine learning models to classify movies into those of high quality which are more likely to be useful and those of poor quality which are likely to prevent a valid comparison between experimental interventions.

6.2 Improved simulation of CPC migration

We designed our simulated movies based on what we observed visually – a general movement of CPCs displaying an arch-like motion from the mid-primitive streak towards the lateral-plate-mesoderm at the anterior of the embryo. We could not use the movies to evaluate our cell detection method as cells in these simulated datasets were also presented as binarized dots, which was because we felt reproducing true CPCs would require time-consuming characterizations of examples of the cells. A more exhaustive approach to generating synthetic data could include establishing ground truths for cell detections and trajectories across frames, this would enable us to better reproduce the visual features cells display as well as simulating their behaviours.

Advances in machine learning may be able to assist in automatically extracting cell features and mechanistic principles and use these to generate unique simulated datasets. Generative Adversarial Networks (GANs) use deep neural networks in a double-model approach – one of which takes example data as an input and outputs a permutated version, while the other acts as a discriminator that classifies the example as either real or synthetic [88]. This approach can be used to generate synthetic datasets with properties similar enough to the original example to falsely be classified as real by the discriminator, thus allowing the creation of unique simulated data. In previous studies, GANs have been used to simulate cell images and movies using images of human embryos [89, 90]. Future studies involving CPC migration could involve applying GANs to generate synthetic datasets of chick embryos to reduce the time and cost involved in producing real time-lapsed-microscopy movies.

6.3 Improved cell detection methods

Our cell detection approach, which involved a Laplacian-style convolutional filter, was selected based on the fact it was the method with the lowest absolute error between true and detected cells in our images. While other methods such as TrackMate's spot detection algorithm may be more generalizable across diverse contexts, it is likely the noise and resolution of our time-lapsed-images were distinct from other contexts. Future studies

Like with image quality auditing, CNNs and other machine learning models could be useful for detecting CPCs in our images, as there are many examples of models trained in other contexts for said purpose such as U-net [91]. While we chose not to apply these models to our images as their training and testing would require abundant and annotated datasets, these annotations may become available in the future as more datasets are generated, requiring more high-throughput methods for cell detection.

6.4 Improved cell tracking methods

In our analysis we abandoned the use of single-cell-tracking in favour of a generalized-tracking of cell groups based on the potential for errors in present in the former methods, such as failing to link correct cells across frames and generating fragmented tracks with gaps between some frames. As generalized-tracking completely avoids these issues, we felt it was a more suitable method for testing our hypotheses that different morphogenic signals can impact CPC migration patterns. This did however limit the scope of our investigation, as we could no longer model the behaviour of individual cells using a generalized approach.

While present single-cell-trackers such as TrackMate were unsuitable for analysing our images, there may be alternative methods of tracking cells as individuals that could generate more complete trajectories with fewer errors. We focused on TrackMate and its respective LAP algorithm as it is one of the more accessible tools available, however there are other single-cell-trackers that were unexplored such as those involving multiple-hypothesis testing [45], active contour modelling [92], the Viterbi algorithm [93] and other graph-based algorithms [94]. We did not exhaustively implement and test all single-cell-tracking methods as we assumed they would present similar issues as displayed by TrackMate, which itself is the most widely published method, however future studies could apply these to ensure this was the case.

As mentioned previously, by annotating images and labelling which cells are to be linked across frames, it is possible to train machine learning models which could predict accurate cell-cell linkages in the absence of any prior information about the mechanistic principles of cell motion. Deep learning models have been implemented for this purpose including DeLTA [95] and 3DeeCellTracker [96], which use CNNs to for both cell detection and tracking. We did not apply pre-trained models for our investigation as their training image resolution and cell shapes displayed little in common with our own context, future investigations however could involve annotating our time-lapsed-microscopy movies to evaluate the performance of advanced machine learning models in generating accurate CPC trajectories.

6.5 Automated feature extraction without tracking

We showed that it is possible to process whole movies into a 2D format that can be used as an input for classifiers ranging from the basic regression models to state-of-the-art vision transformers. With more datasets for training and evaluating these types of models, it may be possible to make comparisons between CPC migration movies without any tracking at all. This would be advantageous, as tracking CPCs in our movie appeared more complex and prone to error than when detecting the cells. It would also allow us to potentially use a larger number of features, as in our analysis and in the *Song et al.* study [6], only the exit angle could be used to reliably compare samples.

Our models made binary predictions based on whether or not the embryos were controls or treated with morphogenic signals. This is fairly limited and in an actual analysis researchers would likely prefer a multiclass classifier or a regression model to represent a larger array of phenotypes. In previous studies using imaging analysis, a multiclass classification model has been used to predict types of skin cancer [97], while others have been used to predict a continuous quantification of COVID-19 infection in lungs [98]. These both present possible future directions for our CPC migration movies or progenitor cells with similar movement patterns.

In terms of accuracy, it could be expected that 2D CNNs would outperform vision transformers given that the latter has been shown to outperform the former only when the sample size approaches a million, which we would not assume the number of CPC migration movies ever to reach. Vision transformers have an interpretability advantage however, as we can visualize what regions of the image the model learns from most [86]. In the future, this could be extremely useful to ensure the model is learning from features that are actually related to the migration of cells rather than noise.

In our analysis we combined a 3D migration TLM into a 2D density map, yet it may be possible to use the entire time-lapses in future studies. CNNs have been used to predict the stage of development up to the 8-cell stage in mouse embryos across a TLM [99]. It is possible a similar approach could be used in our movies, where for each movie we predict the HH stage based on cell positions. An error in the HH stage prediction could be indicative of an aberrant phenotype in embryo development in response to experimental conditions. We were unable to use this approach in our analysis, as the movies frames were not labelled to correspond with their HM stage.

6.6 Conclusion

We identified shortcomings in using TrackMate for both the detection and tracking of CPCs in fluorescent microscopy movies. In the former case, we were able to improve the accuracy of detected CPCs in our images using a custom convolutional filter, which did not improve the cell linkage stage in TrackMate. We instead defined two tracking methods which attempt to build trajectories based on bilateral groups of cells each side of the primitive streak at the middle of the embryo: Frametrack and Mudtrack. Frametrack was more accurate in synthetic data, but generated trajectories which were highly stochastic in actual CPC migration movies. We therefore used Mudtrack for a final analysis comparing GFP-control movies with those overexpressing follistatin, from which we found a significant difference. Furthermore, we developed a framework for comparing CPC migration patterns using cell detections as an input for machine learning models without any need for a tracking algorithm. We suggest that as the number of cell migration movies increases, researchers should consider replacing mechanistic-based cell tracking methods with machine learning models to automatically learn and compare migratory behaviours across different experimental interventions.

Supplementary information

Supplement 1.

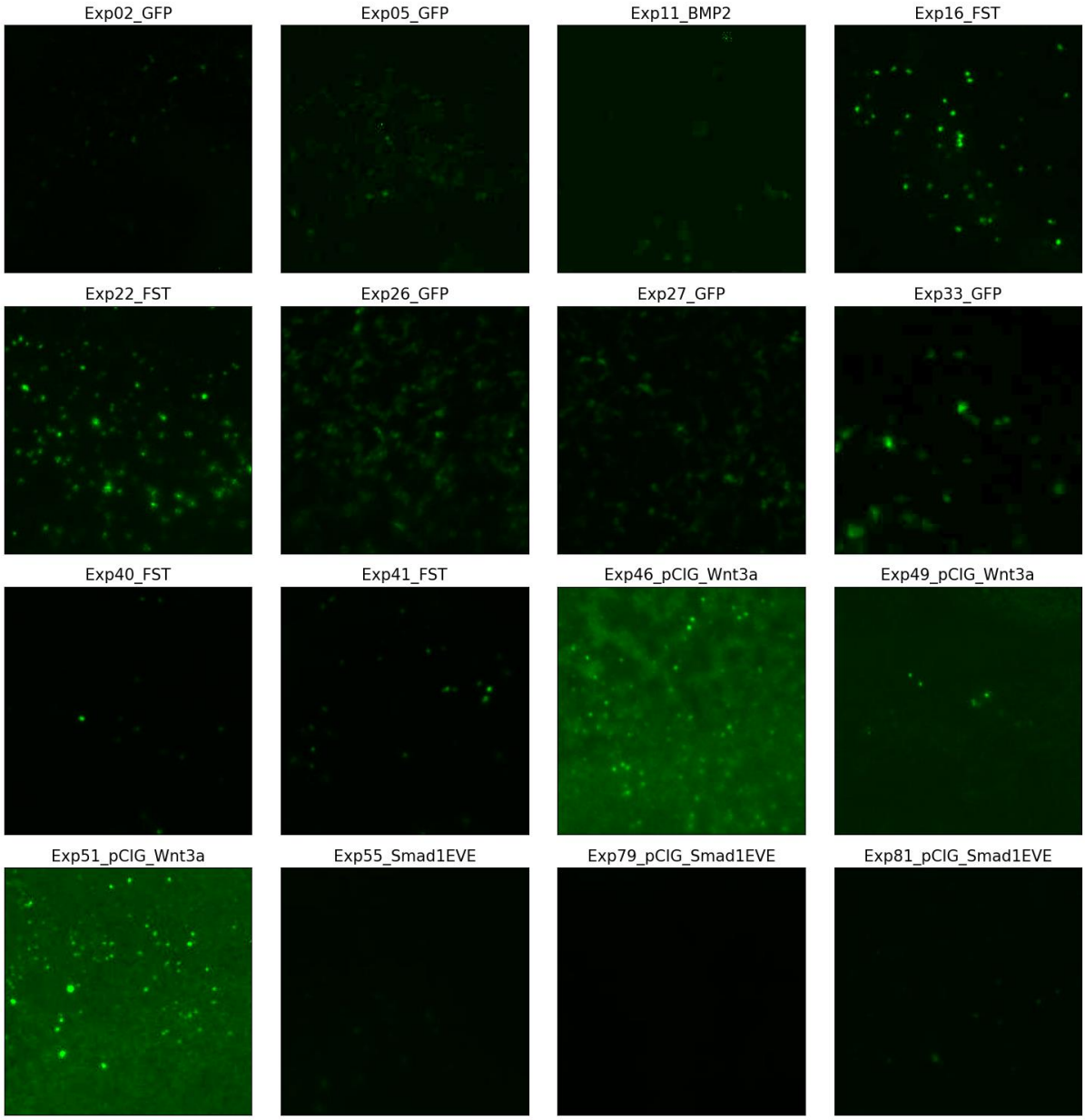
Table S1. All movies with morphogen overexpression and morpholino knockdowns (KD) listed

Name	Overexpression	Morpholino KD
Exp1_GFP	None	None
Exp2_GFP	None	None
Exp3_GFP	None	None
Exp4_GFP	None	None
Exp5_GFP	None	None
Exp6_GFP	None	None
Exp7_GFP	None	None
Exp8_GFP	None	None
Exp9_FST	FST	None
Exp11_BMP2	BMP2	None
Exp12_BMP2	BMP2	None
Exp13_FST	FST	None
Exp16_FST	FST	None
Exp18_FST	FST	None
Exp19_FST	FST	None
Exp20_FST	FST	None
Exp21_FST	FST	None
Exp22_FST	FST	None
Exp23_GFP	None	None
Exp24_GFP	None	None
Exp25_FST	FST	None
Exp26_GFP	None	None
Exp27_GFP	None	None
Exp28_GFP	None	None
Exp29_FST	FST	None
Exp30_FST	FST	None
Exp31_FST	FST	None
Exp32_FST	FST	None
Exp33_GFP	None	None
Exp34_GFP	None	None
Exp35_GFP	None	None
Exp36_GFP	None	None
Exp37_FST	FST	None
Exp38_FST	FST	None

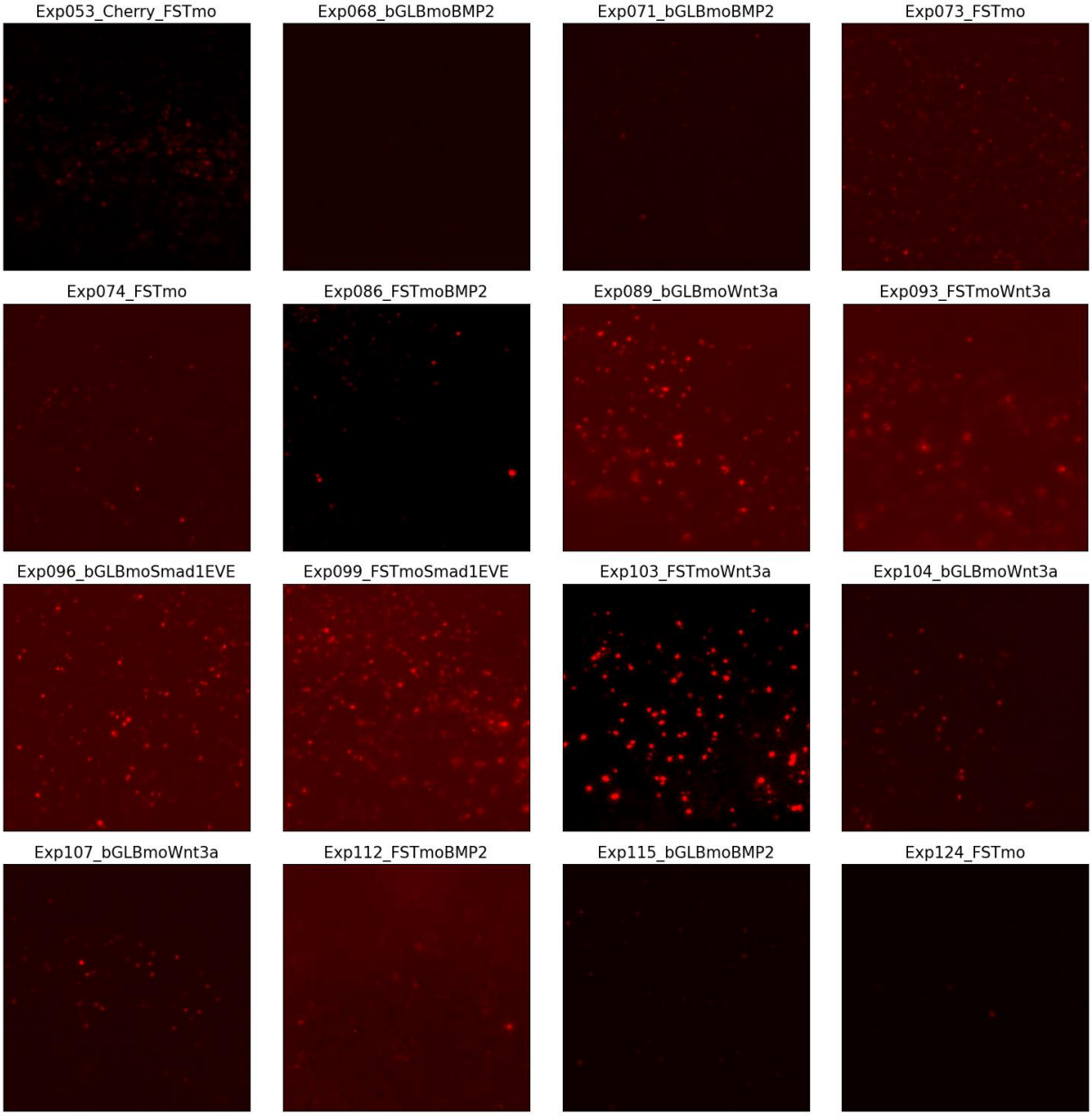
Exp39_FST	**FST**	**None**
Exp40_FST	**FST**	**None**
Exp41_FST	**FST**	**None**
Exp42_FST	**FST**	**None**
Exp43_FST	**FST**	**None**
Exp44_FST	**FST**	**None**
Exp46_pCIG_Wnt3a	**Wnt3a**	**None**
Exp47_pCIG_Wnt3a	**Wnt3a**	**None**
Exp48_pCIG_Wnt3a	**Wnt3a**	**None**
Exp49_pCIG_Wnt3a	**Wnt3a**	**None**
Exp50_pCIG_Wnt3a	**Wnt3a**	**None**
Exp51_pCIG_Wnt3a	**Wnt3a**	**None**
Exp52_pCIG_Wnt3a	**Wnt3a**	**None**
Exp53_Cherry_FSTmo	**None**	**FST**
Exp55_Smad1EVE	**Smad1EVE**	**None**
Exp56_Smad1EVE	**Smad1EVE**	**None**
Exp57_Smad1EVE	**Smad1EVE**	**None**
Exp58_Smad1EVE	**Smad1EVE**	**None**
Exp67_cherry_bGLBmoBMP2	**BMP2**	**bGLB**
Exp67_cherry_bGLBmoBMP22	**BMP22**	**bGLB**
Exp67_cherry_bGLBmoBMP2	**BMP2**	**bGLB**
Exp67_cherry_bGLBmoBMP22	**BMP22**	**bGLB**
Exp68_cherry_bGLBmoBMP2	**BMP2**	**bGLB**
Exp70_cherry_FSTmoBMP2	**BMP2**	**FST**
Exp71_cherry_bGLBmoBMP2	**BMP2**	**bGLB**
Exp72_cherry_bGLBmoBMP2	**BMP2**	**bGLB**
Exp73_cherry_FSTmo	**None**	**FST**
Exp74_cherry_FSTmo	**None**	**FST**
Exp75_cherry_FSTmo	**None**	**FST**
Exp79_pCIG_Smad1EVE	**Smad1EVE**	**None**
Exp80_pCIG_Smad1EVE	**Smad1EVE**	**None**
Exp81_pCIG_Smad1EVE	**Smad1EVE**	**None**
Exp86_cherry_FSTmoBMP2	**BMP2**	**FST**
Exp87_cherry_FSTmoBMP2	**BMP2**	**FST**
Exp88_cherry_FSTmoBMP2	**BMP2**	**FST**
Exp89_cherry_bGLBmoWnt3a	**Wnt3a**	**bGLB**
Exp90_cherry_bGLBmoWnt3a	**Wnt3a**	**bGLB**
Exp91_cherry_FSTmoWnt3a	**Wnt3a**	**FST**
Exp92_cherry_FSTmoWnt3a	**Wnt3a**	**FST**
Exp93_cherry_FSTmoWnt3a	**Wnt3a**	**FST**
Exp94_cherry_FSTmoWnt3a	**Wnt3a**	**FST**

Exp95_cherry_bGLBmoSmad1Eve	**Smad1Eve**	**bGLB**
Exp96_cherry_bGLBmoSmad1EVE	**Smad1EVE**	**bGLB**
Exp97_cherry_bGLBmoSmad1EVE	**Smad1EVE**	**bGLB**
Exp98_cherry_FSTmoSmad1EVE	**Smad1EVE**	**FST**
Exp99_cherry_FSTmoSmad1EVE	**Smad1EVE**	**FST**
Exp100_cherry_FSTmoSmad1EVE	**Smad1EVE**	**FST**
Exp101_cherry_FSTmoWnt3a	**Wnt3a**	**FST**
Exp102_cherry_FSTmoWnt3a	**Wnt3a**	**FST**
Exp103_cherry_FSTmoWnt3a	**Wnt3a**	**FST**
Exp104_cherry_bGLBmoWnt3a	**Wnt3a**	**bGLB**
Exp105_cherry_bGLBmoWnt3a	**Wnt3a**	**bGLB**
Exp106_cherry_bGLBmoWnt3a	**Wnt3a**	**bGLB**
Exp107_cherry_bGLBmoWnt3a	**Wnt3a**	**bGLB**
Exp110_cherry_bGLBmoBMP2	**BMP2**	**bGLB**
Exp111_cherry_FSTmo	**None**	**FST**
Exp112_cherry_FSTmoBMP2	**BMP2**	**FST**
Exp113_cherry_bGLBmoBMP2	**BMP2**	**bGLB**
Exp114_cherry_bGLBmoBMP2	**BMP2**	**bGLB**
Exp115_cherry_bGLBmoBMP2	**BMP2**	**bGLB**
Exp116_cherry_bGLBmoBMP2	**BMP2**	**bGLB**
Exp117_cherry_FSTmoBMP2	**BMP2**	**FST**
Exp118_cherry_FSTmoBMP2	**BMP2**	**FST**
Exp119_cherry_FSTmo	**None**	**FST**
Exp121_cherry_FSTmo	**None**	**FST**
Exp122_cherry_FSTmo	**None**	**FST**
Exp123_cherry_FSTmo	**None**	**FST**
Exp124_cherry_FSTmo	**None**	**FST**
Exp125_cherry_FSTmo	**None**	**FST**
Exp126_cherry_FSTmo	**None**	**FST**

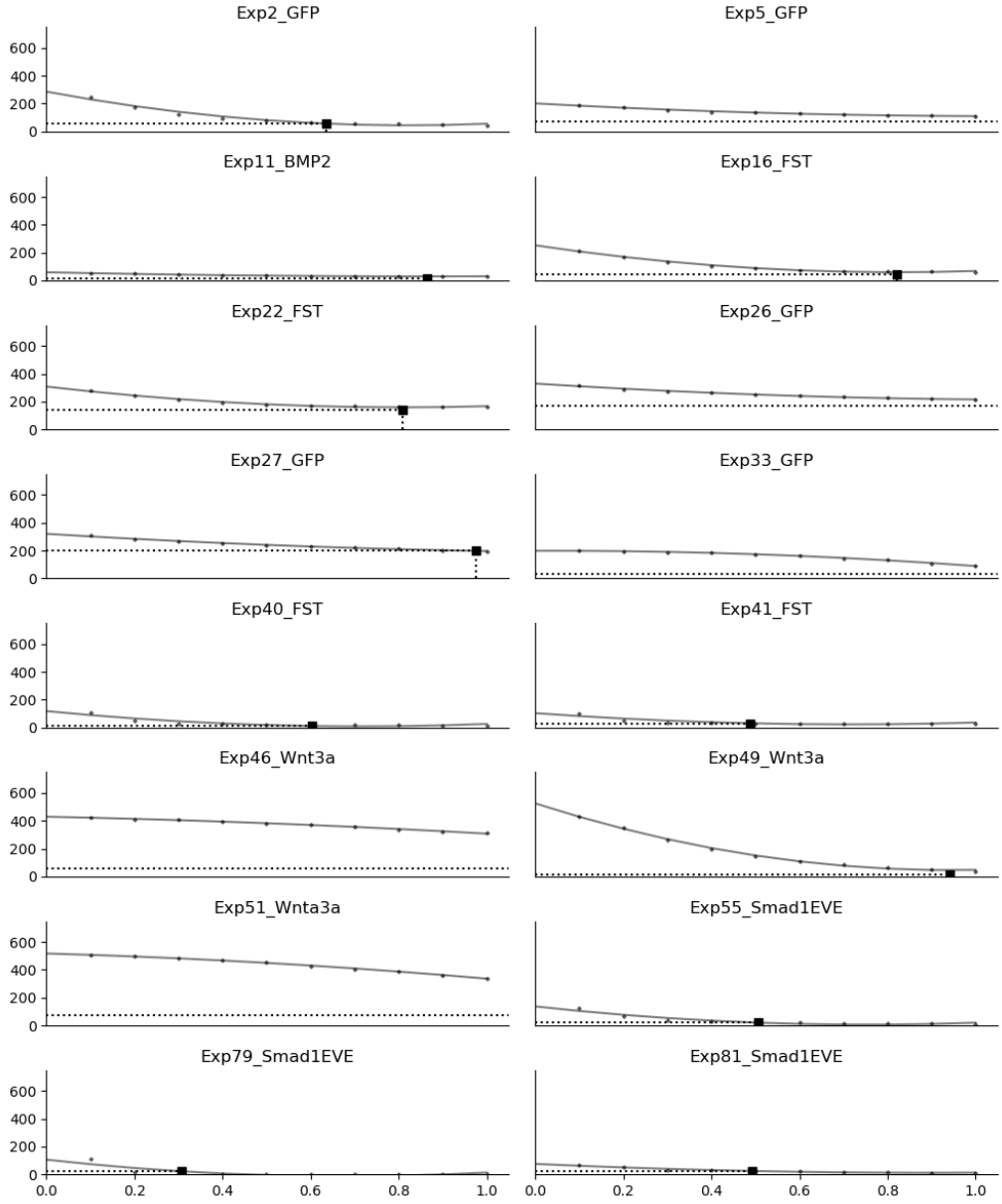
Supplement 2.



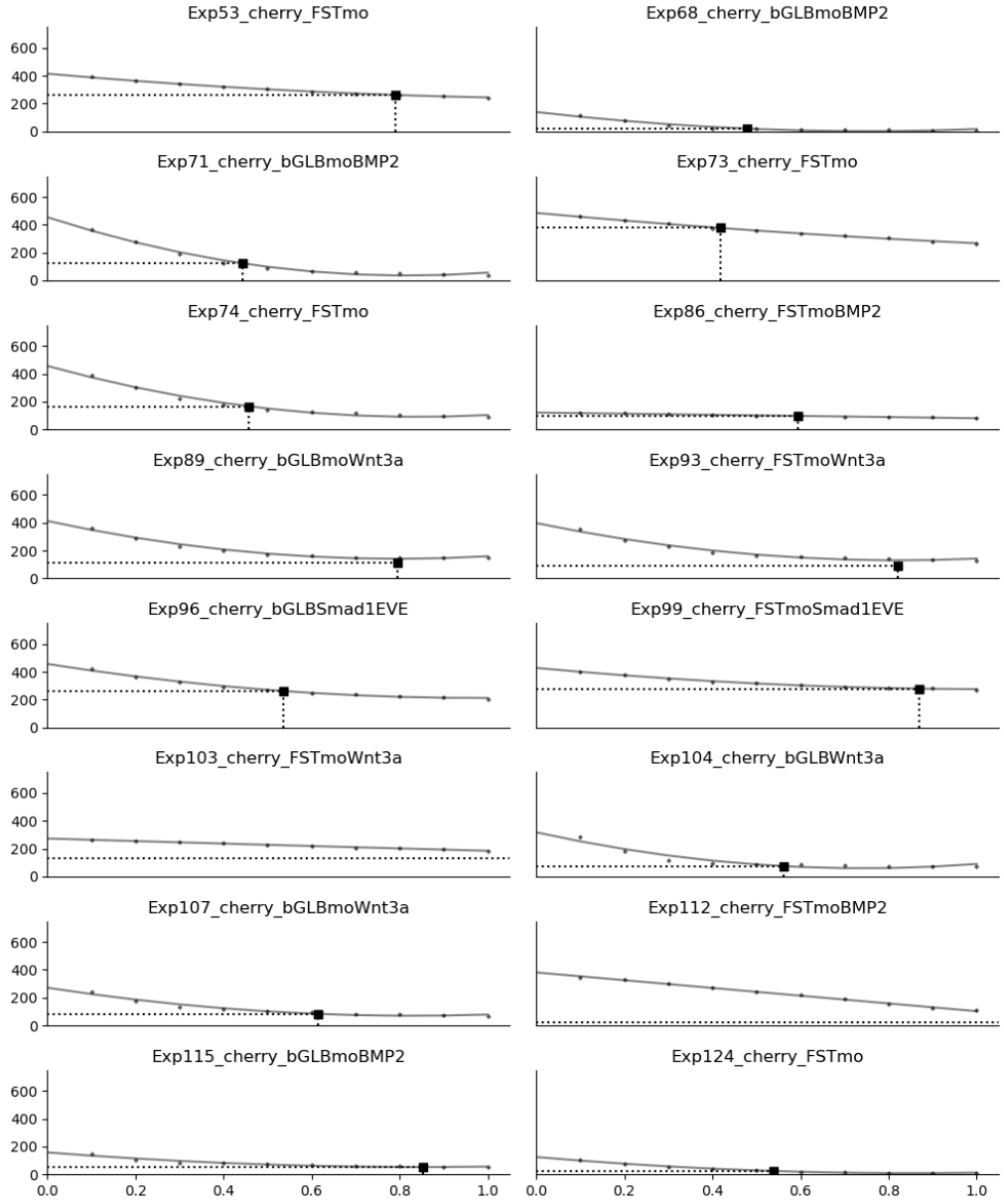
S2.1 A sample of 16 200x200 partitions taken from early embryonic development TLM-movies in *gallus gallus* with GFP expressing CPCs.



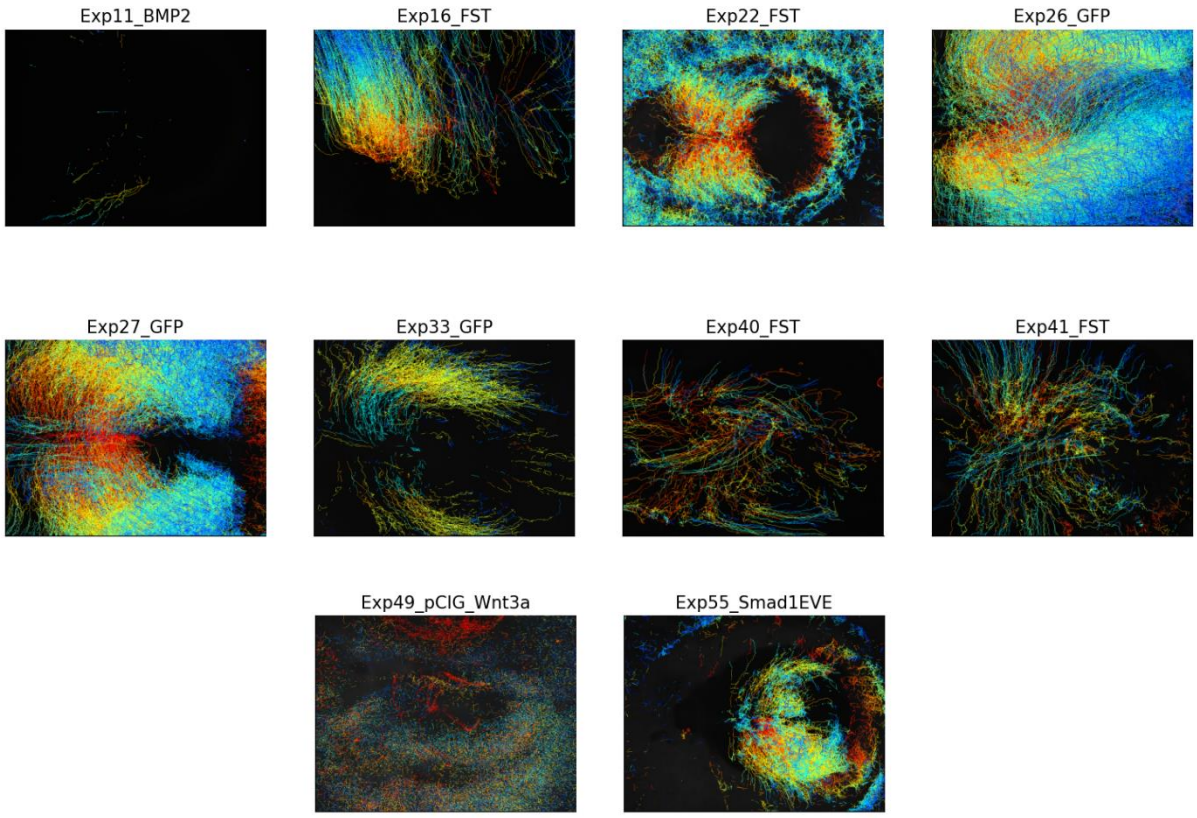
S2.2 A sample of 16 200x200 partitions taken from early embryonic development TLM-movies in *gallus gallus* with Cherry expressing CPCs.



S2.3 The change in cell count detections in GFP expressing image partitions across different user-defined values for quality thresholding using TrackMate's LoG spot detector. 2nd degree polynomial regression models were fitted, and a ground truth cell count (spotted line) was used to predict the ideal quality threshold for the image.

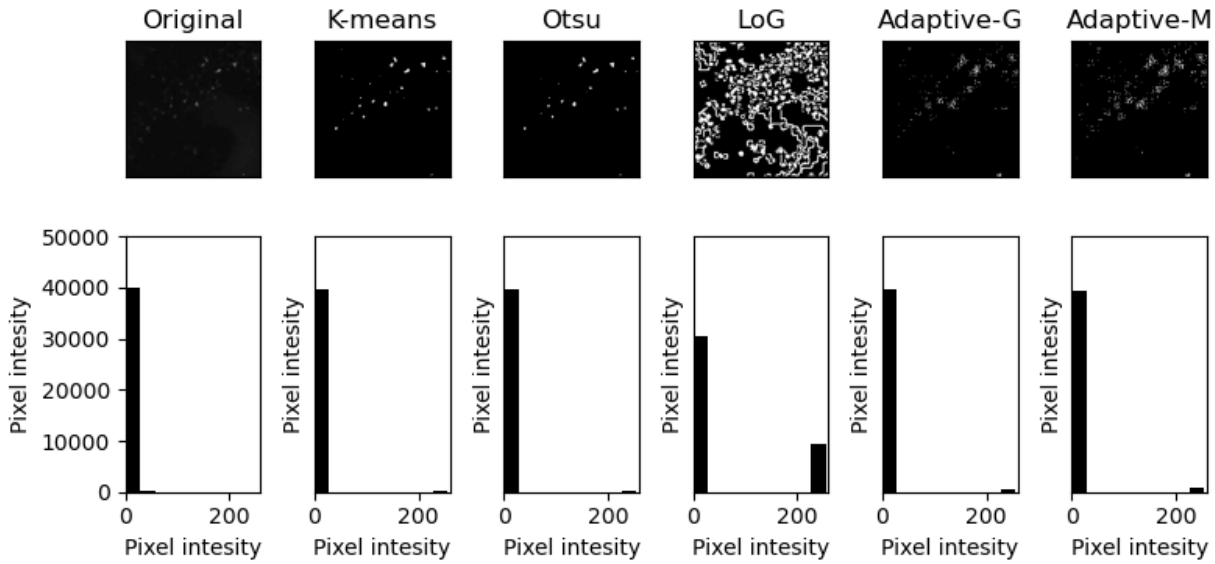


S2.4 T the change in cell count detections in Cherry expressing image partitions across different user-defined values for quality thresholding using TrackMate's LoG spot detector. 2nd degree polynomial regression models were fitted, and a ground truth cell count (spotted line) was used to predict the ideal quality threshold for the image.



S2.6 Visualizations of TrackMate's LAP spot tracker linkages using predicted ideal threshold values in GFP expressing movies. Colour gradient flows from red to blue proportion to time.

Supplement 3.

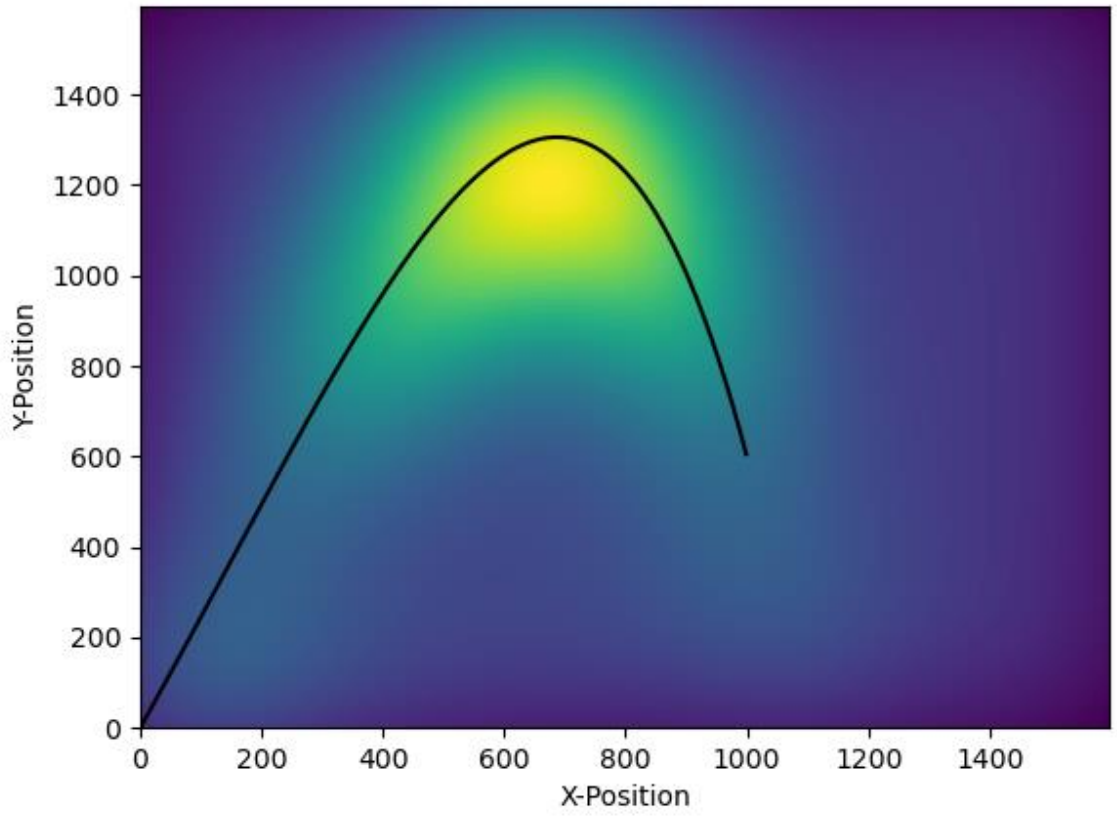


S3.1 Image partitions in unprocessed as well as K-means, Otsu, LoG, Adaptive Gaussian and Adaptive median segmented images (top) with their pixel intensity distributions (bottom).

Code for cell detection methods:

https://github.com/ConnorReynoldsUK/MScRes_thesis/blob/main/Chapter3_SEGMENTATION.ipynb

Supplement 4.



S4.1 Density map of synthetic movie generated with the original function shown in black.

Code for Simulated movies:

https://github.com/ConnorReynoldsUK/MScRes_thesis/blob/main/Chapter4_simulation.ipynb

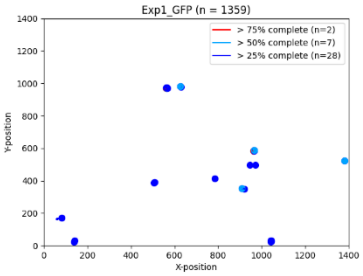
Code for Frametrack method:

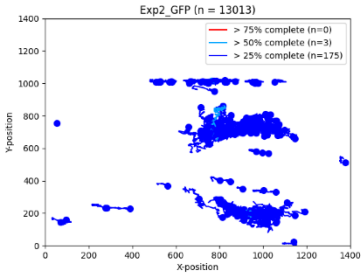
https://github.com/ConnorReynoldsUK/MScRes_thesis/blob/main/Chapter4_Frametrack.ipynb

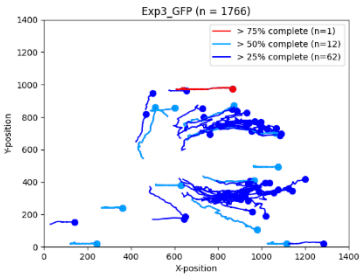
Code for Mudtrack method:

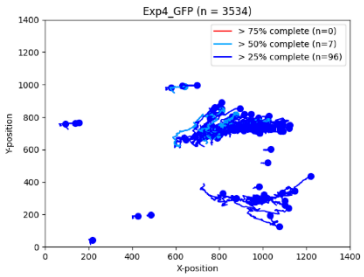
https://github.com/ConnorReynoldsUK/MScRes_thesis/blob/main/Chapter4_Mudtrack.ipynb

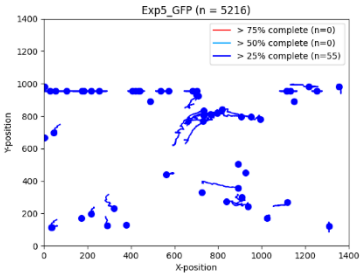
Supplement 5.

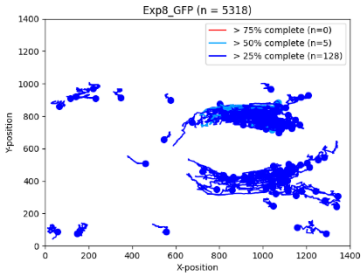


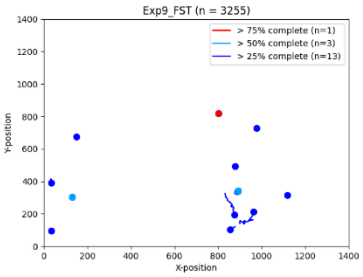


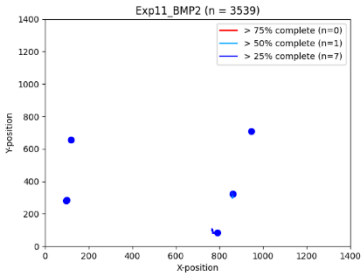


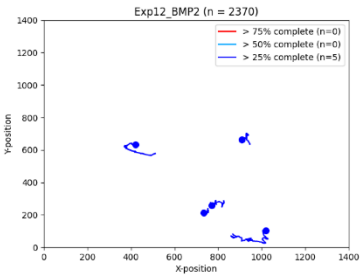


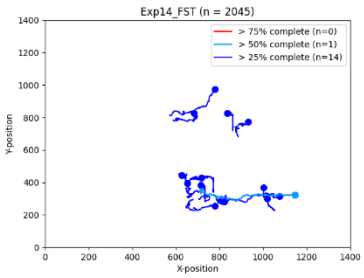


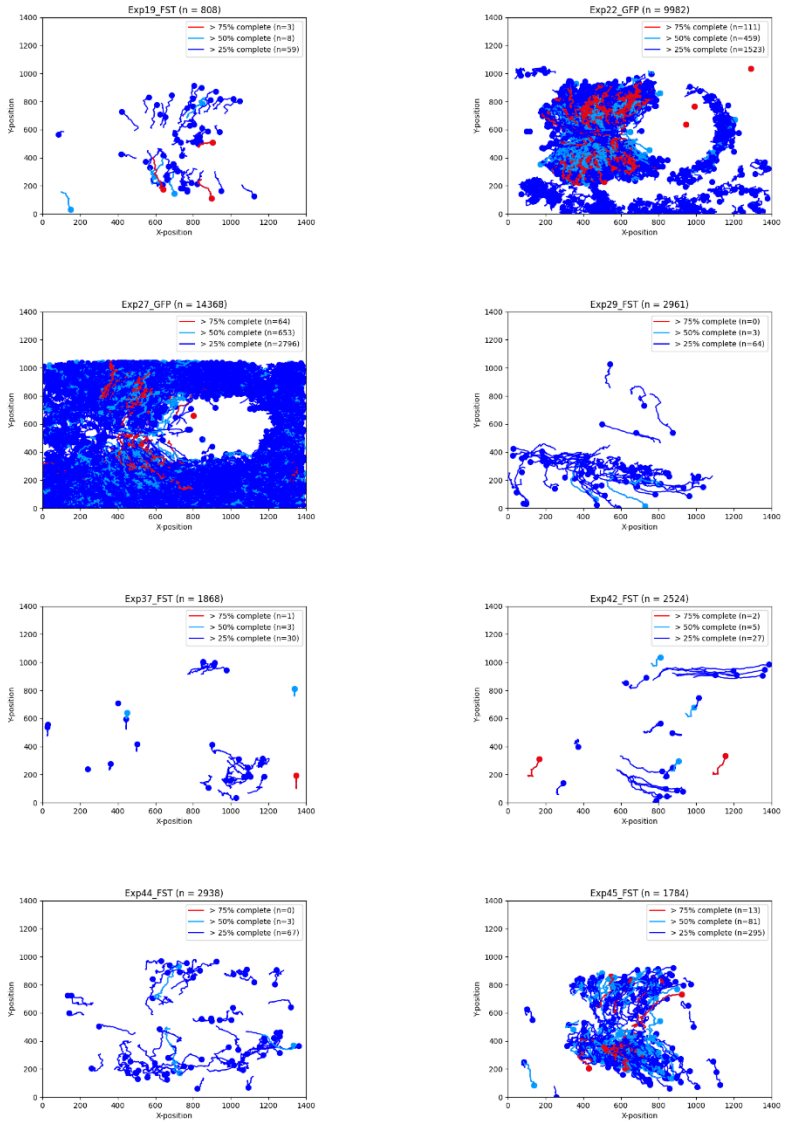




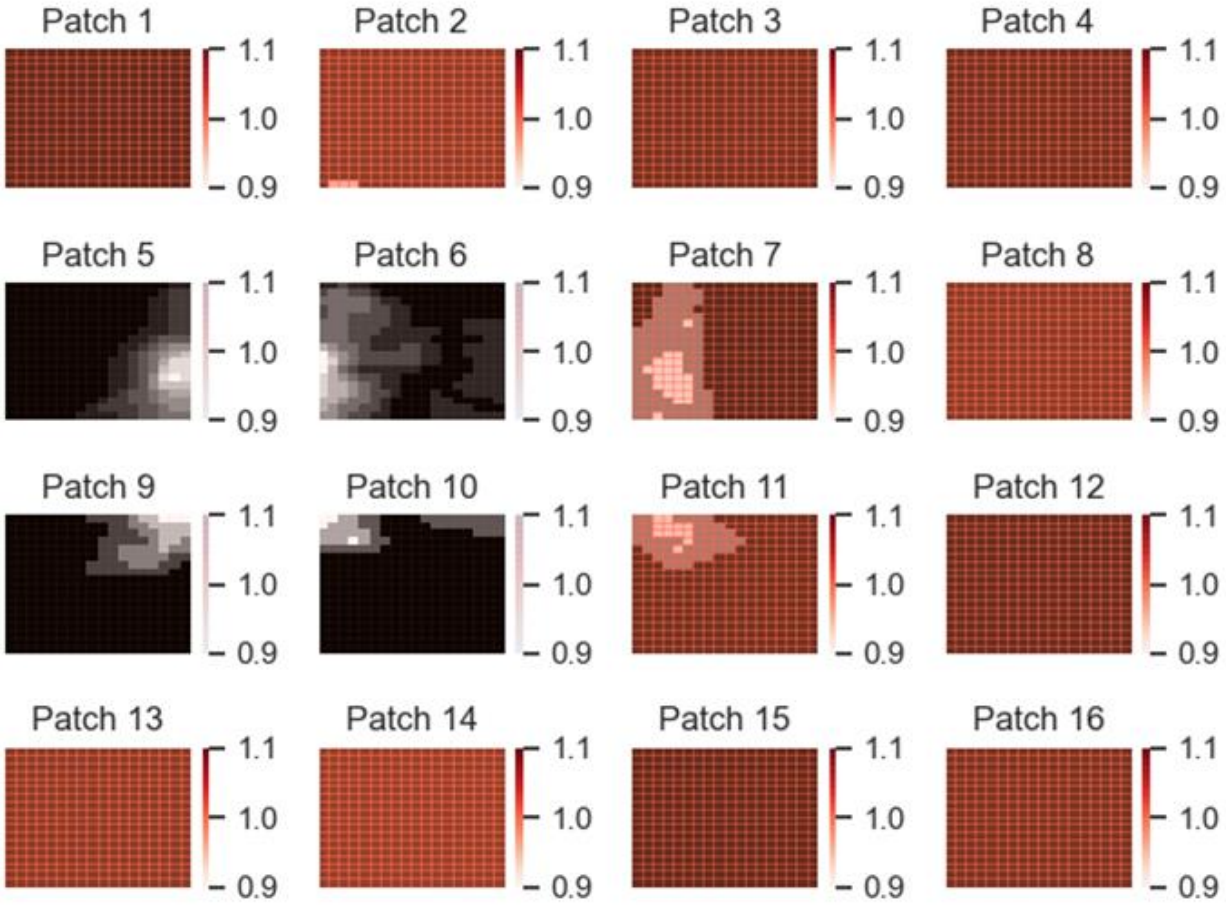








S5.1 Trajectories generated in TrackMate including those with >75% coverage (red), >50% coverage (cyan) and >25% coverage (blue)



S5.2 Patches of density map with attention scores overlayed. Higher attention scores shown in red while lower shown in grayscale.

Table S5.1 2D CNN model layers and parameters

Layer (type)	Output shape	Parameters #
conv2d	(None, 73, 73, 64)	640
conv2d_1	(None, 71, 71, 64)	36928
dropout	(None, 71, 71, 64)	0
max_pooling2d	(None, 35, 35, 64)	0
flatten	(None, 78400)	0
dense	(None, 256)	20070656
dense_1	(None, 2)	514

References

1. Voronov DA, Alford PW, Xu G, Taber LA (2004) The role of mechanical forces in dextral rotation during cardiac looping in the chick embryo. Dev Biol 272:339–350. https://doi.org/10.1016/j.ydbio.2004.04.033

2. Schoenwolf GC, Garcia-Martinez V (1995) Primitive-streak origin and state of commitment of cells of the cardiovascular system in avian and mammalian embryos. Cell Mol Biol Res 41:233–240

3. Stalsberg H, DeHaan RL (1969) The precardiac areas and formation of the tubular heart in the chick embryo. Developmental Biology 19:128–159. https://doi.org/10.1016/0012-1606(69)90052-9

4. Colas JF, Lawson A, Schoenwolf GC (2000) Evidence that translation of smooth muscle alpha-actin mRNA is delayed in the chick promyocardium until fusion of the bilateral heart-forming regions. Dev Dyn 218:316–330. https://doi.org/10.1002/(SICI)1097-0177(200006)218:2<316::AID-DVDY6>3.0.CO;2-8

5. Yue Q, Wagstaff L, Yang X, Weijer C, Münsterberg A (2008) Wnt3a-mediated chemorepulsion controls movement patterns of cardiac progenitors and requires RhoA function. Development 135:1029–1037. https://doi.org/10.1242/dev.015321

6. Song J, McColl J, Camp E, Kennerley N, Mok GF, McCormick D, Grocott T, Wheeler GN, Münsterberg AE (2014) Smad1 transcription factor integrates BMP2 and Wnt3a signals in migrating cardiac progenitor cells. PNAS 111:7337–7342. https://doi.org/10.1073/pnas.1321764111

7. Gräper L (1929) Die Primitiventwicklung des Hühnchens nach stereokinematographischen Untersuchungen, kontrolliert durch vitale Farbmarkierung und verglichen mit der Entwicklung anderer Wirbeltiere. Wilhelm Roux Arch Entwickl Mech Org 116:382–429. https://doi.org/10.1007/BF02145235

8. Wetzel R (1929) Untersuchungen am Hühnchen. Die Entwicklung des Keims während der ersten beiden Bruttage. Wilhelm Roux Arch Entwickl Mech Org 119:188–321. https://doi.org/10.1007/BF02111186

9. Waddington CH (1933) Induction by the endoderm in birds. Wilhelm Roux Arch Entwickl Mech Org 128:502–521. https://doi.org/10.1007/BF00649862

10. Chapman SC, Collignon J, Schoenwolf GC, Lumsden A (2001) Improved method for chick whole-embryo culture using a filter paper carrier. Dev Dyn 220:284–289. https://doi.org/10.1002/1097-0177(20010301)220:3<284::AID-DVDY1102>3.0.CO;2-5

11. Dormann D, Weijer CJ (2006) Imaging of cell migration. EMBO J 25:3480–3493. https://doi.org/10.1038/sj.emboj.7601227

12. Chuai M, Hughes D, Weijer CJ (2012) Collective Epithelial and Mesenchymal Cell Migration During Gastrulation. Curr Genomics 13:267–277. https://doi.org/10.2174/138920212800793357

13. Hamburger V, Hamilton HL (1951) A series of normal stages in the development of the chick embryo. J Morphol 88:49–92

14. Martinsen BJ (2005) Reference guide to the stages of chick heart embryology. Dev Dyn 233:1217–1237. https://doi.org/10.1002/dvdy.20468

15. Gittenberger-de Groot AC, Bartelings MM, Deruiter MC, Poelmann RE (2005) Basics of cardiac development for the understanding of congenital heart malformations. Pediatr Res 57:169–176. https://doi.org/10.1203/01.PDR.0000148710.69159.61

16. Hatada Y, Stern CD (1994) A fate map of the epiblast of the early chick embryo. Development 120:2879–2889. https://doi.org/10.1242/dev.120.10.2879

17. Yatskievych TA, Ladd AN, Antin PB (1997) Induction of cardiac myogenesis in avian pregastrula epiblast: the role of the hypoblast and activin. Development 124:2561–2570. https://doi.org/10.1242/dev.124.13.2561

18. Yang X, Dormann D, Münsterberg AE, Weijer CJ (2002) Cell movement patterns during gastrulation in the chick are controlled by positive and negative chemotaxis mediated by FGF4 and FGF8. Dev Cell 3:425–437. https://doi.org/10.1016/s1534-5807(02)00256-3

19. Marvin MJ, Di Rocco G, Gardiner A, Bush SM, Lassar AB (2001) Inhibition of Wnt activity induces heart formation from posterior mesoderm. Genes Dev 15:316–327. https://doi.org/10.1101/gad.855501

20. Schultheiss TM, Burch JB, Lassar AB (1997) A role for bone morphogenetic proteins in the induction of cardiac myogenesis. Genes Dev 11:451–462. https://doi.org/10.1101/gad.11.4.451

21. Ladd AN, Yatskievych TA, Antin PB (1998) Regulation of avian cardiac myogenesis by activin/TGFbeta and bone morphogenetic proteins. Dev Biol 204:407–419. https://doi.org/10.1006/dbio.1998.9094

22. Osmond MK, Butler AJ, Voon FC, Bellairs R (1991) The effects of retinoic acid on heart formation in the early chick embryo. Development 113:1405–1417

23. Osei-Sarfo K, Gudas LJ (2014) Retinoic acid suppresses the canonical Wnt signaling pathway in embryonic stem cells and activates the noncanonical Wnt signaling pathway. Stem Cells 32:2061–2071. https://doi.org/10.1002/stem.1706

24. Dehaan RL (1963) Migration patterns of the precardiac mesoderm in the early chick embrvo. Exp Cell Res 29:544–560. https://doi.org/10.1016/s0014-4827(63)80016-6

25. Hamilton N (2009) Quantification and its applications in fluorescent microscopy imaging. Traffic 10:951–961. https://doi.org/10.1111/j.1600-0854.2009.00938.x

26. Obara B, Roberts MAJ, Armitage JP, Grau V (2013) Bacterial cell identification in differential interference contrast microscopy images. BMC Bioinformatics 14:134. https://doi.org/10.1186/1471-2105-14-134

27. Fan H, Sakamoto N, Aoyama H (2018) From the primitive streak to the somitic mesoderm: labeling the early stages of chick embryos using EGFP transfection. Anat Sci Int 93:414–421. https://doi.org/10.1007/s12565-018-0429-y

28. McColl J, Mok GF, Lippert AH, Ponjavic A, Muresan L, Münsterberg A (2018) 4D imaging reveals stage dependent random and directed cell motion during somite morphogenesis. Scientific Reports 8:12644. https://doi.org/10.1038/s41598-018-31014-3

29. Lopez-Schier H, Pinto-Teixeira F, Muzzopappa M, Swoger J, Mineo A, Sharpe J (2013) Intravital imaging of hair-cell development and regeneration in the zebrafish. Frontiers in Neuroanatomy 7:33. https://doi.org/10.3389/fnana.2013.00033

30. Myers CT, Krieg PA (2013) BMP-mediated specification of the erythroid lineage suppresses endothelial development in blood island precursors. Blood 122:3929–3939. https://doi.org/10.1182/blood-2013-03-490045

31. Ulman V, Maška M, Magnusson KEG, Ronneberger O, Haubold C, Harder N, Matula P, Matula P, Svoboda D, Radojevic M, Smal I, Rohr K, Jaldén J, Blau HM, Dzyubachyk O, Lelieveldt B, Xiao P, Li Y, Cho S-Y, Dufour AC, Olivo-Marin J-C, Reyes-Aldasoro CC, Solis-Lemus JA, Bensch R, Brox T, Stegmaier J, Mikut R, Wolf S, Hamprecht FredA, Esteves T, Quelhas P, Demirel Ö, Malmström L, Jug F, Tomancak P, Meijering E, Muñoz-Barrutia A, Kozubek M, Ortiz-de-Solorzano C (2017) An Objective Comparison of Cell Tracking Algorithms. Nat Methods 14:1141–1152. https://doi.org/10.1038/nmeth.4473

32. Chenouard N, Smal I, de Chaumont F, Maška M, Sbalzarini IF, Gong Y, Cardinale J, Carthel C, Coraluppi S, Winter M, Cohen AR, Godinez WJ, Rohr K, Kalaidzidis Y, Liang L, Duncan J, Shen H, Xu Y, Magnusson KEG, Jaldén J, Blau HM, Paul-Gilloteaux P, Roudot P, Kervrann C, Waharte F, Tinevez J-Y, Shorte SL, Willemse J, Celler K, van Wezel GP, Dan H-W, Tsai Y-S, de Solórzano CO, Olivo-Marin J-C, Meijering E (2014) Objective comparison of particle tracking methods. Nat Methods 11:281–289. https://doi.org/10.1038/nmeth.2808

33. Tanenbaum ME, Gilbert LA, Qi LS, Weissman JS, Vale RD (2014) A Protein-Tagging System for Signal Amplification in Gene Expression and Fluorescence Imaging. Cell 159:635–646. https://doi.org/10.1016/j.cell.2014.09.039

34. Carpenter AE, Kamentsky L, Eliceiri KW (2012) A call for bioimaging software usability. Nat Methods 9:666–670. https://doi.org/10.1038/nmeth.2073

35. Imaris for Tracking - Imaris. In: Oxford Instruments. https://imaris.oxinst.com/products/imaris-for-tracking. Accessed 7 Dec 2021

36. Sacan A, Ferhatosmanoglu H, Coskun H (2008) CellTrack: an open-source software for cell tracking and motility analysis. Bioinformatics 24:1647–1649. https://doi.org/10.1093/bioinformatics/btn247

37. Held M, Schmitz MHA, Fischer B, Walter T, Neumann B, Olma MH, Peter M, Ellenberg J, Gerlich DW (2010) CellCognition: time-resolved phenotype annotation in high-throughput live cell imaging. Nat Methods 7:747–754. https://doi.org/10.1038/nmeth.1486

38. Huth J, Buchholz M, Kraus JM, Mølhave K, Gradinaru C, Wichert G v., Gress TM, Neumann H, Kestler HA (2011) TimeLapseAnalyzer: Multi-target analysis for live-cell imaging and time-lapse microscopy. Computer Methods and Programs in Biomedicine 104:227–234. https://doi.org/10.1016/j.cmpb.2011.06.002

39. Klein J, Leupold S, Biegler I, Biedendieck R, Münch R, Jahn D (2012) TLM-Tracker: software for cell segmentation, tracking and lineage analysis in time-lapse microscopy movies. Bioinformatics 28:2276–2277. https://doi.org/10.1093/bioinformatics/bts424

40. Cooper S, Barr AR, Glen R, Bakal C (2017) NucliTrack: an integrated nuclei tracking application. Bioinformatics 33:3320–3322. https://doi.org/10.1093/bioinformatics/btx404

41. Tinevez J-Y, Perry N, Schindelin J, Hoopes GM, Reynolds GD, Laplantine E, Bednarek SY, Shorte SL, Eliceiri KW (2017) TrackMate: An open and extensible platform for single-particle tracking. Methods 115:80–90. https://doi.org/10.1016/j.ymeth.2016.09.016

42. McQuin C, Goodman A, Chernyshev V, Kamentsky L, Cimini BA, Karhohs KW, Doan M, Ding L, Rafelski SM, Thirstrup D, Wiegraebe W, Singh S, Becker T, Caicedo JC, Carpenter AE (2018) CellProfiler 3.0: Next-generation image processing for biology. PLoS Biol 16:e2005970. https://doi.org/10.1371/journal.pbio.2005970

43. de Chaumont F, Dallongeville S, Chenouard N, Hervé N, Pop S, Provoost T, Meas-Yedid V, Pankajakshan P, Lecomte T, Le Montagner Y, Lagache T, Dufour A, Olivo-Marin J-C (2012) Icy: an open bioimage informatics platform for extended reproducible research. Nat Methods 9:690–696. https://doi.org/10.1038/nmeth.2075

44. Jaqaman K, Loerke D, Mettlen M, Kuwata H, Grinstein S, Schmid SL, Danuser G (2008) Robust single particle tracking in live cell time-lapse sequences. Nat Methods 5:695–702. https://doi.org/10.1038/nmeth.1237

45. Chenouard N, Bloch I, Olivo-Marin J-C (2013) Multiple Hypothesis Tracking for Cluttered Biological Image Sequences. IEEE Transactions on Pattern Analysis and Machine Intelligence 35:2736–3750. https://doi.org/10.1109/TPAMI.2013.97

46. Hilsenbeck O, Schwarzfischer M, Skylaki S, Schauberger B, Hoppe PS, Loeffler D, Kokkaliaris KD, Hastreiter S, Skylaki E, Filipczyk A, Strasser M, Buggenthin F, Feigelman JS, Krumsiek J, van den Berg AJJ, Endele M, Etzrodt M, Marr C, Theis FJ, Schroeder T (2016) Software tools for single-cell tracking and quantification of cellular and molecular properties. Nat Biotechnol 34:703–706. https://doi.org/10.1038/nbt.3626

47. Welcome to Python.org. In: Python.org. https://www.python.org/. Accessed 27 Jun 2022

48. NumPy. https://numpy.org/. Accessed 27 Jun 2022

49. scikit-learn: machine learning in Python — scikit-learn 1.1.1 documentation. https://scikit-learn.org/stable/. Accessed 27 Jun 2022

50. Home. In: OpenCV. https://opencv.org/. Accessed 10 Feb 2021

51. Schroeder AB, Dobson ETA, Rueden CT, Tomancak P, Jug F, Eliceiri KW (2021) The ImageJ ecosystem: Open-source software for image visualization, processing, and analysis. Protein Science 30:234–249. https://doi.org/10.1002/pro.3993

52. Graham JE, Marians KJ, Kowalczykowski SC (2017) Independent and Stochastic Action of DNA Polymerases in the Replisome. Cell 169:1201-1213.e17. https://doi.org/10.1016/j.cell.2017.05.041

53. Owen LM, Adhikari AS, Patel M, Grimmer P, Leijnse N, Kim MC, Notbohm J, Franck C, Dunn AR (2017) A cytoskeletal clutch mediates cellular force transmission in a soft, three-dimensional extracellular matrix. MBoC 28:1959–1974. https://doi.org/10.1091/mbc.e17-02-0102

54. Fazeli E, Roy NH, Follain G, Laine RF, von Chamier L, Hänninen PE, Eriksson JE, Tinevez J-Y, Jacquemet G (2020) Automated cell tracking using StarDist and TrackMate. F1000Res 9:1279. https://doi.org/10.12688/f1000research.27019.1

55. Lowe DG (2004) Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision 60:91–110. https://doi.org/10.1023/B:VISI.0000029664.99615.94

56. Munkres J (1957) Algorithms for the Assignment and Transportation Problems. Journal of the Society for Industrial and Applied Mathematics 5:32–38

57. Kass M, Witkin A, Terzopoulos D (1988) Snakes: Active contour models. Int J Comput Vision 1:321–331. https://doi.org/10.1007/BF00133570

58. Cui C, Yang X, Chuai M, Glazier JA, Weijer CJ (2005) Analysis of tissue flow patterns during primitive streak formation in the chick embryo. Dev Biol 284:37–47. https://doi.org/10.1016/j.ydbio.2005.04.021

59. Li X, Zhang T, Shen X, Sun J (2010) Object tracking using an adaptive Kalman filter combined with mean shift. OE 49:020503. https://doi.org/10.1117/1.3327281

60. Wuttisarnwattana P, Gargesha M, Hof W van't, Cooke KR, Wilson DL (2016) Automatic Stem Cell Detection in Microscopic Whole Mouse Cryo-imaging. IEEE Trans Med Imaging 35:819–829. https://doi.org/10.1109/TMI.2015.2497285

61. Ghaye J, Kamat MA, Corbino-Giunta L, Silacci P, Vergères G, De Micheli G, Carrara S (2013) Image thresholding techniques for localization of sub-resolution fluorescent biomarkers. Cytometry A 83:1001–1016. https://doi.org/10.1002/cyto.a.22345

62. Malpica N, de Solórzano CO, Vaquero JJ, Santos A, Vallcorba I, García-Sagredo JM, del Pozo F (1997) Applying watershed algorithms to the segmentation of clustered nuclei. Cytometry 28:289–297. https://doi.org/10.1002/(sici)1097-0320(19970801)28:4<289::aid-cyto3>3.0.co;2-7

63. Sadanandan SK, Ranefall P, Le Guyader S, Wählby C (2017) Automated Training of Deep Convolutional Neural Networks for Cell Segmentation. Sci Rep 7:7860. https://doi.org/10.1038/s41598-017-07599-6

64. Elboher E, Werman M (2011) Efficient and Accurate Gaussian Image Filtering Using Running Sums. arXiv:11074958 [cs]

65. Babic ZV, Mandic DP (2003) An efficient noise removal and edge preserving convolution filter. In: 6th International Conference on Telecommunications in Modern Satellite, Cable and Broadcasting Service, 2003. TELSIKS 2003. pp 538–541 vol.2

66. Niemistö A, Selinummi J, Saleem R, Shmulevich I, Aitchison J, Yli-Harja O (2006) Extraction of the number of peroxisomes in yeast cells by automated image analysis. Conf Proc IEEE Eng Med Biol Soc 2006:2353–2356. https://doi.org/10.1109/IEMBS.2006.259890

67. Kong H, Akakin HC, Sarma SE (2013) A generalized Laplacian of Gaussian filter for blob detection and its applications. IEEE Trans Cybern 43:1719–1733. https://doi.org/10.1109/TSMCB.2012.2228639

68. Liu D, Yu J (2009) Otsu Method and K-means. In: 2009 Ninth International Conference on Hybrid Intelligent Systems. pp 344–349

69. Du X, Dua S (2010) Segmentation of fluorescence microscopy cell images using unsupervised mining. Open Med Inform J 4:41–49. https://doi.org/10.2174/1874431101004020041

70. Shen SP, Tseng H, Hansen KR, Wu R, Gritton HJ, Si J, Han X (2018) Automatic Cell Segmentation by Adaptive Thresholding (ACSAT) for Large-Scale Calcium Imaging Datasets. eNeuro 5:. https://doi.org/10.1523/ENEURO.0056-18.2018

71. Haralick RM, Sternberg SR, Zhuang X (1987) Image Analysis Using Mathematical Morphology. IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-9:532–550. https://doi.org/10.1109/TPAMI.1987.4767941

72. Di Rubeto C, Dempster A, Khan S, Jarra B (2000) Segmentation of blood images using morphological operators. In: Proceedings 15th International Conference on Pattern Recognition. ICPR-2000. pp 397–400 vol.3

73. Kornilov AS, Safonov IV (2018) An Overview of Watershed Algorithm Implementations in Open Source Libraries. Journal of Imaging 4:123. https://doi.org/10.3390/jimaging4100123

74. Salem N, Sobhy NM, Dosoky ME (2016) A Comparative Study of White Blood cells Segmentation using Otsu Threshold and Watershed Transformation. Journal of Biomedical Engineering and Medical Imaging 3:15–15. https://doi.org/10.14738/jbemi.33.2078

75. Meijering E, Dzyubachyk O, Smal I (2012) Chapter nine - Methods for Cell and Particle Tracking. In: conn PM (ed) Methods in Enzymology. Academic Press, pp 183–200

76. Molnar C, Jermyn IH, Kato Z, Rahkama V, Östling P, Mikkonen P, Pietiäinen V, Horvath P (2016) Accurate Morphology Preserving Segmentation of Overlapping Cells based on Active Contours. Sci Rep 6:32412. https://doi.org/10.1038/srep32412

77. Fukunaga K, Hostetler L (1975) The estimation of the gradient of a density function, with applications in pattern recognition. IEEE Transactions on Information Theory 21:32–40. https://doi.org/10.1109/TIT.1975.1055330

78. Karavasilis V, Nikou C, Likas A (2010) Visual Tracking by Adaptive Kalman Filtering and Mean Shift. In: Konstantopoulos S, Perantonis S, Karkaletsis V, Spyropoulos CD, Vouros G (eds) Artificial Intelligence: Theories, Models and Applications. Springer, Berlin, Heidelberg, pp 153–162

79. Xiang-Zhang, Dai Y-M, Chen Z-W, Zhang H-X (2010) An improved Mean Shift tracking algorithm based on color and texture feature. In: 2010 International Conference on Wavelet Analysis and Pattern Recognition. pp 38–43

80. Iswanto IA, Li B (2017) Visual Object Tracking Based on Mean-shift and Particle-Kalman Filter. Procedia Computer Science 116:587–595. https://doi.org/10.1016/j.procs.2017.10.010

81. Tokunaga T, Hirose O, Kawaguchi S, Toyoshima Y, Teramoto T, Ikebata H, Kuge S, Ishihara T, Iino Y, Yoshida R (2014) Automated detection and tracking of many cells by using 4D live-cell imaging data. Bioinformatics 30:i43–i51. https://doi.org/10.1093/bioinformatics/btu271

82. Debeir O, Van Ham P, Kiss R, Decaestecker C (2005) Tracking of migrating cells under phase-contrast video microscopy with combined mean-shift processes. IEEE Transactions on Medical Imaging 24:697–711. https://doi.org/10.1109/TMI.2005.846851

83. SciPy.org — SciPy.org. https://www.scipy.org/. Accessed 27 Jun 2021

84. pykalman — pykalman 0.9.2 documentation. https://pykalman.github.io/#. Accessed 15 Aug 2021

85. Machine learning education | TensorFlow. https://www.tensorflow.org/resources/learn-ml?gclid=CjwKCAjwquWVBhBrEiwAt1Kmws7pPSqHnd4Ulk1VxOci1VQOICQaOqM-7jrS3LKc3XbjKIbkG6fjhBoC7UIQAvD_BwE. Accessed 27 Jun 2022

86. Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, Dehghani M, Minderer M, Heigold G, Gelly S, Uszkoreit J, Houlsby N (2021) An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. arXiv

87. (2022) Keras.io documentation generator. Keras

88. Yi X, Walia E, Babyn P (2019) Generative adversarial network in medical imaging: A review. Medical Image Analysis 58:101552. https://doi.org/10.1016/j.media.2019.101552

89. Comes MC, Filippi J, Mencattini A, Casti P, Cerrato G, Sauvat A, Vacchelli E, De Ninno A, Di Giuseppe D, D'Orazio M, Mattei F, Schiavoni G, Businaro L, Di Natale C, Kroemer G, Martinelli E (2021) Multi-scale generative adversarial network for improved evaluation of cell–cell interactions observed in organ-on-chip experiments. Neural Comput & Applic 33:3671–3689. https://doi.org/10.1007/s00521-020-05226-6

90. Dirvanauskas D, Maskeliūnas R, Raudonis V, Damaševičius R, Scherer R (2019) HEMIGEN: Human Embryo Image Generator Based on Generative Adversarial Networks. Sensors (Basel) 19:3578. https://doi.org/10.3390/s19163578

91. Falk T, Mai D, Bensch R, Çiçek Ö, Abdulkadir A, Marrakchi Y, Böhm A, Deubner J, Jäckel Z, Seiwald K, Dovzhenko A, Tietz O, Dal Bosco C, Walsh S, Saltukoglu D, Tay TL, Prinz M, Palme K, Simons M, Diester I, Brox T, Ronneberger O (2019) U-Net: deep learning for cell counting, detection, and morphometry. Nat Methods 16:67–70. https://doi.org/10.1038/s41592-018-0261-2

92. Huang Y, Liu Z (2015) Segmentation and Tracking of Lymphocytes Based on Modified Active Contour Models in Phase Contrast Microscopy Images. Computational and Mathematical Methods in Medicine 2015:e693484. https://doi.org/10.1155/2015/693484

93. Magnusson KEG, Jalden J, Gilbert PM, Blau HM (2015) Global linking of cell tracks using the Viterbi algorithm. IEEE Trans Med Imaging 34:911–929. https://doi.org/10.1109/TMI.2014.2370951

94. Löffler K, Scherr T, Mikut R (2021) A graph-based cell tracking algorithm with few manually tunable parameters and automated segmentation error correction. PLOS ONE 16:e0249257. https://doi.org/10.1371/journal.pone.0249257

95. Lugagne J-B, Lin H, Dunlop MJ (2020) DeLTA: Automated cell segmentation, tracking, and lineage reconstruction using deep learning. PLOS Computational Biology 16:e1007673. https://doi.org/10.1371/journal.pcbi.1007673

96. Wen C, Miura T, Voleti V, Yamaguchi K, Tsutsumi M, Yamamoto K, Otomo K, Fujie Y, Teramoto T, Ishihara T, Aoki K, Nemoto T, Hillman EM, Kimura KD (2021) 3DeeCellTracker, a deep learning-based pipeline for segmenting and tracking cells in 3D time lapse images. eLife 10:e59187. https://doi.org/10.7554/eLife.59187

97. Maron RC, Weichenthal M, Utikal JS, Hekler A, Berking C, Hauschild A, Enk AH, Haferkamp S, Klode J, Schadendorf D, Jansen P, Holland-Letz T, Schilling B, von Kalle C, Fröhling S, Gaiser MR, Hartmann D, Gesierich A, Kähler KC, Wehkamp U, Karoglan A, Bär C, Brinker TJ, Schmitt L, Peitsch WK, Hoffmann F, Becker JC, Drusio C, Jansen P, Klode J, Lodde G, Sammet S, Schadendorf D, Sondermann W, Ugurel S, Zader J, Enk A, Salzmann M, Schäfer S, Schäkel K, Winkler J, Wölbing P, Asper H, Bohne A-S, Brown V, Burba B, Deffaa S, Dietrich C, Dietrich M, Drerup KA, Egberts F, Erkens A-S, Greven S, Harde V, Jost M, Kaeding M, Kosova K, Lischner S, Maagk M, Messinger AL, Metzner M, Motamedi R, Rosenthal A-C, Seidl U, Stemmermann J, Torz K, Velez JG, Haiduk J, Alter M, Bär C, Bergenthal P, Gerlach A, Holtorf C, Karoglan A, Kindermann S, Kraas L, Felcht M, Gaiser MR, Klemke C-D, Kurzen H, Leibing T, Müller V, Reinhard RR, Utikal J, Winter F, Berking C, Eicher L, Hartmann D, Heppt M, Kilian K, Krammer S, Lill D, Niesert A-C, Oppel E, Sattler E, Senner S, Wallmichrath J, Wolff H, Giner T, Glutsch V, Kerstan A, Presser D, Schrüfer P, Schummer P, Stolze I, Weber J, Drexler K, Haferkamp S, Mickler M, Stauner CT, Thiem A (2019) Systematic outperformance of 112 dermatologists in multiclass skin cancer image classification

by convolutional neural networks. European Journal of Cancer 119:57–65. https://doi.org/10.1016/j.ejca.2019.06.013

98. Zhang H, Zhang J, Zhang H, Nan Y, Zhao Y, Fu E, Xie Y, Liu W, Li W, Zhang H, Jiang H, Li C, Li Y, Ma R, Dang S, Gao B, Zhang X, Zhang T (2020) Automated detection and quantification of COVID-19 pneumonia: CT imaging analysis by a deep learning-based software. Eur J Nucl Med Mol Imaging 47:2525–2532. https://doi.org/10.1007/s00259-020-04953-1

99. Malmsten J, Zaninovic N, Zhan Q, Toschi M, Rosenwaks Z, Shan J (2018) Automatic prediction of embryo cell stages using artificial intelligence convolutional neural network. Fertility and Sterility 110:e360. https://doi.org/10.1016/j.fertnstert.2018.07.1005