Tracking transient fluorescent events in structured point clouds

Saoirse Amarteifio[1]*, Todd Fallesen[2], Giovanni Sena[2], Gunnar Pruessner[1]

1 Department of Mathematics, Imperial College London
2 Department of Life Sciences, Imperial College London

* s.amarteifio14@imperial.ac.uk

Abstract

Tracking algorithms used in conjunction with fluorescent time-lapse microscopy data typically assume a *continuous* signal where (a) background and keypoints are permanently visible and (b) independently moving objects of interest are permanently visible when not occluded. These conditions allow for registration and *identity management* algorithms to track independently-moving objects of interest over time. In contrast to such conditions, we consider the case of (a) *transient* fluorescent events and (b) objects embedded in a possibly imperceptible, *almost rigid structure*, which acts to constrain independent object movement. In a biological context, such events could for example represent cell divisions in a growing tissue. Such conditions motivate the merging of registration and tracking tasks into a fuzzy registration algorithm to solve the identity management problem. We describe the design and application of such an algorithm, illustrated in the domain of plant biology and made available as an open source software implementation.

Introduction

It is now generally acknowledged that automated computer vision and tracking methods can match or exceed manual human-led tracking of objects in large image datasets and that due to the rate at which such data are being produced, these methods are of growing importance [1, 2].

Fluorescence time-lapse microscopy exploits markers that emit light continuously for a period of time allowing for objects of interest to be tracked [1, 3–6] notwithstanding occlusions or objects leaving and entering scenes. Here we consider *transient* fluorescent events where objects in the field of view fluoresce for a relatively short time to mark *events of interest*. While our model is general, the case-study considered here uses transient fluorescent markers to detect mitotic events in growing *Arabidopsis* roots. To count distinct mitotic events using transient fluorescent markers we effectively exchange the more common segmentation with morphological analysis problem [6] for a fuzzy registration-tracking problem.

Point-set registration [7–10] involves identifying *keypoint* correspondences in two frames. Keypoints may correspond for example to distinctive edges or blobs that exist in both frames. If points appear in only one frame they are treated as *outliers* in the registration problem. Given a keypoint correspondence (pairs of *inliers*), an affine transformation (*i.e.* a generalised transformation including translations, rotations, scalings) that takes one set of points to the other can be determined. Our data typically

contain few guaranteed inliers as fluorescence events typically have short life-times (several frames) and tissue structures may be difficult to identify. This is the key problem that we have tried to address. The problem should be understood as sitting between a single particle tracking problem and a registration problem. Single particle tracking solutions may or may not require pre-registration. In the case that they do require pre-registration, keypoint registration solutions may not be robust against lack of guaranteed inliners. In the case where single particle tracking methods are robust in the absence of pre-registration, they typically emphasise individual object motion models, which may not be optimal under certain data conditions *i.e.* transient events in rigid structures and low signal to noise ratio across extended images sequences.

Object *tracking* in time-lapse image data involves *detection* of objects in individual frames and solving the *data association* or *identity management* problem between frames. The goal of identity management is to link an object observed at one time point to what is found to be the same object observed at a later time point. Unique *identifiers* are propagated between married objects, *i.e.* objects determined to be the same object in each frame. The set of observations of a given object through time is referred to as it's *lineage*. One early example of a tracking algorithm, *multiple hypothesis tracking* [11, 12] takes an exhaustive, deterministic approach to consider possible lineage trees. Other examples may take a stochastic approach [13] which can be more efficient and robust in the context of noisy data. Tracking approaches are greatly influenced by the nature of the data. For tracking real-life scenes, objects may be complex and may be distinguished on high-level image features, which can enhance tracking [14]. In fluorescence time-lapse cell microscopy [5] objects may be virtually indistinguishable such that tracking relies more strongly on *motion models* alone. Myriad and diverse examples of tracking applications are discussed in the literature as seen for example in review articles such as [2, 15].

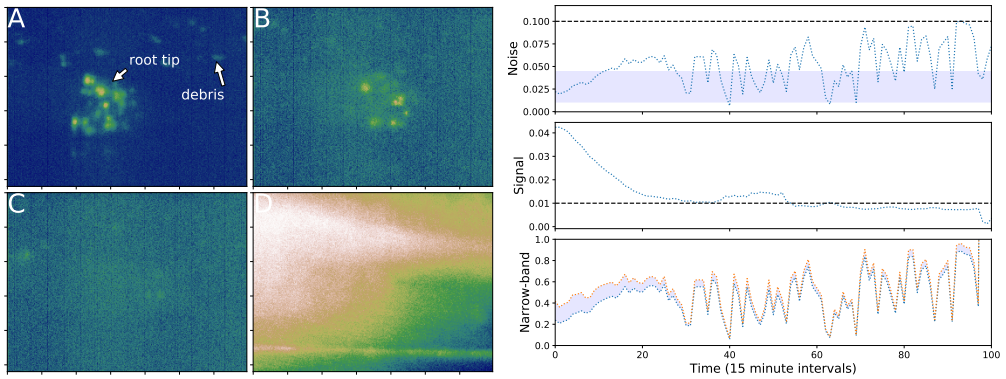
The core of the article can be found in Section 2 where we describe the general problem and our solution to tracking fluorescent events in noisy point cloud data. We discuss some of the data preprocessing steps specific to our data in the next section. The algorithm is further evaluated and discussed in Section 3. The fully automated Python software has been made available on GitHub via the provided link [16]. It can be easily installed over a Python scientific library distribution such as *Anaconda*. It includes interactive *notebooks* and can be run from a terminal window to process a folder containing 3D image sequence data in fully automated fashion.

1 Methods

1.1 Data acquisition

A single *Arabidopsis* primary root is grown and imaged on a custom-made light-sheet microscope setup, as previously described [17, 18]. In essence, the root is hydroponically grown in a *perfusion chamber* maintained under constant light and constant temperature, with its liquid medium fully exchanged every 2 minutes. A full 3D scan in fluorescence of the root tip is composed of 60 optical sections 4μm apart, captured every 15 minutes for up to 7 days. To visualize mitotic events, we use an established transgenic line expressing a fusion between the cyclin protein CYCB1;1 and the fluorescent protein GFP [19]. In these plants, the fluorescent reporter CYCB1;1::GFP accumulates in a cell transitioning between G2 and M phases of the cell cycle, and is quickly degraded after entering mitosis. It is widely adopted as a reliable live marker for cell events.

In our study, we accurately track and count such division events in a temporal series of 3D images.



(a) **(b)**

Fig 1. Variation in frame properties: The tiles left(a) show four sample frames. Frame A is a "good" image. Frame B is a noisy image. Frame C is a noisy image with a low number of objects and some noise. Frame D is over-saturated and will be marked as degenerate. On the right(b), the same story is told quantitatively over 100 consecutive sample frames from one imaging session. The **signal** and **noise** levels are plotted with threshold values. The **narrow-band** plot shows the region between the 95^{th} and 99^{th} percentile ranges of the image histogram. Notably, these values fluctuate dramatically.

1.2 Preprocessing

1.2.1 Noise analysis

Noise and histogram levels are used on normalised grey-scale images taking pixel values in $[0, 1]$. The noise shown in the top panel of Fig. 1B is estimated using a wavelet-based estimator of the Gaussian noise standard deviation [20, 21]. The middle panel in Fig. 1B describes the signal. This is a simple quasi-signal metric defined as a ratio between the 99^{th} and 95^{th} histogram percentile boundary values. If this value approaches 0 we find there is low signal/information. For values of this measure above a calibrated threshold, we find the data are more concentrated outside the 99^{th} percentile of the data, corresponding to bright singularities in the image. A noise range is used to guide downstream strategies in the image processing pipeline. A normal noise band $[0.01, 0.045]$ is used in the sample parameters discussed here. If the noise is very low e.g below 0.01, no denoising is required otherwise wavelet denoising will be applied. Excessive noise, exceeding 0.045, suggests light saturation and/or low signal-to-noise ratio typically due to a low number of biological events. In such instances additional thresholding is applied to the data before wavelet denoising. If the noise exceeds 0.1, the signal-to-noise ratio is so low that the frame is marked as degenerate. Fig. 1A contrasts different noise levels in sample images and Fig. 1B plots image properties over time for a given experiment.

1.2.2 Region of interest

Bright blobs are scattered across the full image and may correspond to mitotic events within the root or debris floating beyond the root. For tracking accuracy and tracking speed, it is advisable to select a *region of interest* (ROI), drawing a box around the identified root.

A 2D projection of the data is obtained by summing the 3D tensor along the z-axis. An adaptive threshold based on image histogram percentile ranges is used to construct a narrow-band filter for the 2D data. This range can vary erratically between frames as shown in the bottom time series in Fig. 1B. A thin slice of the image data range

(shaded band) is selected and an aggressive "Gaussian smoothing" (averaging neighbouring pixel values using a Gaussian kernel) with a large sigma value is used to find a mesh-like connected component corresponding to the region of activity in the root tip. The 2D mask of this largest component is extended to a 3D mask by projecting the 2D region back into the z-plane. The largest connected component in the thresholded image is identified as the root tip. These stages are illustrated in Fig. 2. The image processing pipeline continues to process data only within the ROI.

For cases where the region of activity is well-isolated (smaller region of interest), the processing is more efficient as the amount of volumetric data is reduced and debris beyond the root are filtered from the downstream pipeline. For our sample data, a ROI with area $\approx 10^5$ pixels corresponds to a well-isolated root tip. Noise and light saturation can affect this part of the process making the ROI area a proxy for image quality. While the actual area will depend on the data, it should vary smoothly and be relatively small unless root activity genuinely extends to the entire field of view.

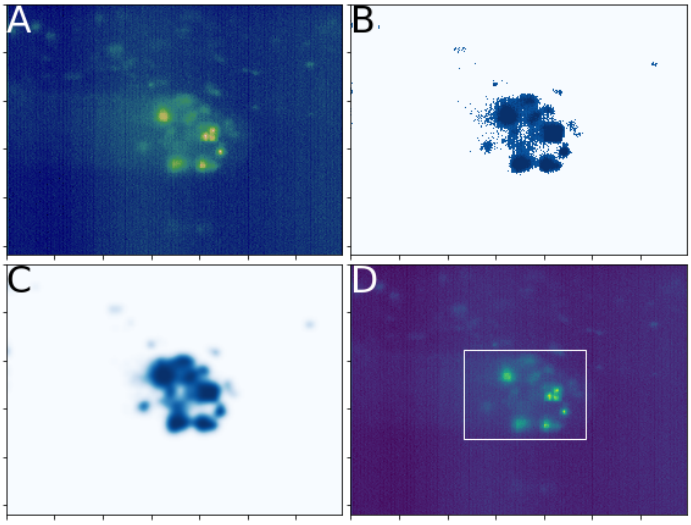


Fig 2. Isolating the Region of Interest (ROI): A narrow-band filter (B) and aggressive Gaussian smoothing (C) is used to find the largest connected component in the narrow band of the data corresponding to the region of activity in the root tip (D).

1.2.3 Object detection

We perform a "pre-segmentation" procedure to detect blob centroids. Segmentation plays a central role in many image processing pipelines and typically involves (i) thresholding and identifying background, (ii) using distance/gradient transformations with peak detection to identify markers and (iii) routines such as watershed [22] to segment blob labels. For our data, we have found it appropriate to not carry out the final segmentation. Instead we carry out the pre-segmentation steps from thresholding to peak detection in detecting object centroids.

Given the variability in the data over a large frame sequence we have found a simple "annealing thresholding" to be effective. This simply increases a threshold iteratively so as to remove large connected components beyond a maximum perimeter length thus removing the background from the image. Being iterative, this is not as efficient as simpler linear filters but it is simple and robust to noise and variability. In extreme cases where it is not possible to find a threshold level in this manner, we treat the 99^{th} percentile as image background and remove it. Having applied the threshold filter, we

identify centroids by (i) performing a difference of Gaussians to emphasise blob-like objects, (ii) applying a maximum filter and (iii) returning the coordinates of the local maxima (peaks) in the image. To allow for a fully automated routine that can cope with arbitrary datasets (in the scope of our light-sheet microscopy datasets) the emphasis in the centroid detection stage has been to avoid spurious centroid detections at the risk of under-sampling, while optimising for objects to be identified for at least two frames somewhere during the peak of their light intensity arc.

2 Tracking Algorithm

The position, appearance and disappearance of transient objects will be treated as a random spatial process. The frame-to-frame displacement of objects will be described as a sum of three random variables: (1) the global movement of the tissue within the field of view, (2) the global movement due the growth of the tissue and (3) small fluctuations of the object (cell) within the tissue. The main purpose of the algorithm is to filter the global movement.

The strategy of the algorithm is to find point set correspondences between the random process at time t and the random process at time $t-\tau$ where τ is a lag variable. Objects that appear in both frames are *inliers* and those that do not appear in both frames are *outliers*. Outliers can be either *debris* i.e. any object which is determined *not* to be an object of interest or could be an object of interest that has "exited" or "entered" the new frame. From the image processing stage, any object outside the bounding box will be an outlier. More generally, an outlier is *an object that cannot be explained by a global frame-frame transformation*.

There are two conceptual phases of the algorithm which may be mixed during optimisation; (i) proposal transformations are generated from the data using one or more strategies and (ii) the optimal transformation with respect to some objective is chosen as the global transformation. Inliers are those objects which can be mapped to each other by the global transformation. Identifiers are propagated between the inliers in each frame in the *linkage* phase. New identifiers are generated for outliers within the region of interest. These algorithm stages are discussed in more detail in the following subsections. Algorithm listing 1 provides a terse, high-level overview of the stages that will be discussed.

2.1 Proposing transformations

Generally, a RANSAC strategy iteratively extracts a random sample from all data and partitions inliers from outliers in the data sample [23]. Consider sets of points $U(t), V(t-\tau)$ at times offset by a lag τ. A subset of these points corresponds to true cells (i.e. excluding noise and debris) and a subset of these true cells will exist in both frames (due to short-lived events). There are N points at time t and M points at time $t-\tau$. A number of permutations $\mathcal{P}(N,k)$ are sampled from U and a number of permutations $\mathcal{P}(M,k)$ are sampled from V. In our method $k \in \{1,3\}$, thus sampling either single points or triangular *constellations* from each frame. Points take a *natural order* and the same points can only appear in one constellation, as illustrated in Fig. 3 and Fig. 4.

Let n be the number of k-constellations $c_i(t)$ sampled from $U(t)$ and m be the number of k-constellations $c_j(t')$ sampled from $V(t-\tau)$. In general, for two tensors $\mathbf{M}_1$ and $\mathbf{M}_2$ related by $\mathbf{M}_1 = \mathbf{A}\mathbf{M}_2$, the affine transformation $\mathbf{A}$ can be uniquely determined through a least squares method. The transformation γ_{ij} is the affine transformation, taking the point set $c_i(t)$ to the point set $c_j(t')$.

Algorithm 1 Transient point cloud tracking algorithm

High-level description of algorithm using 1-indexing
*Standard geometry or data structure functions are *lower camel case**
Other FUNCTIONS are explained in the main text
*Key tensor *sets* in **bold font** shorthand are understood from function calls*

1: $results := \{\}$
2: **for** $i \leftarrow lag + 1, frameCount$ **do**
3: $U := frames(i)$
4: $V := frames(i - lag)$
5: $\mathbf{uv} := cartesianPointProduct(U, V)$
6: $\mathbf{tr1} := translationsFor(\mathbf{uv})$ ▷ For points, k=1
7: $ranked := \text{RANK}(\mathbf{tr1}, U, V, epsilon)$
8: $linkages := \text{APPLY}(ranked, U, V, epsilon)$
9: $\mathbf{kuv} := \text{CONSTELLATIONS}(U, V, linkages, k)$ ▷ constellation pairs
10: $\mathbf{trk} := transformsFor(\mathbf{kuv})$
11: $other := \text{PRIORS}(linkages, U, V)$
12: $transforms := concatenate(\mathbf{tr1}, \mathbf{trk}, other)$

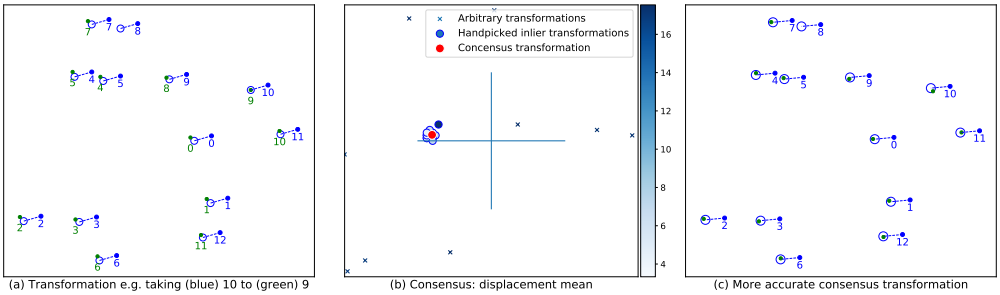
13: $ranked := \text{RANK}(transforms, U, V, epsilon)$
14: $result := \text{APPLY}(ranked, U, V, epsilon)$
15: $results := concatenate(results, result)$

16: $results \leftarrow \text{MINFRAMEDETECTIONFILTERING}(result, minFrames)$
17: $results \leftarrow \text{ROIBELIEFFILTERING}(result)$
18: $results \leftarrow \text{ANISOTROPICCATCHMENTFILTERING}(result)$

Translation sampling Referring to Algorithm listing 1, on lines 5 and 6 constellations are sampled from the data in frames U and V and translations are computed. Sampling returns pairs of points. For comparison with other stages, we think of points as *k-constellations* with $k = 1$. As seen in Fig. 3A, translations are generated for each pair of points. The translation taking a point in $v \in V$ to a point $u \in U$ is added to a list of candidate transformations.

Preliminary linkages While translations and all general affine transformations could be generated in a single phase, it is more efficient to evaluate the linkages found from translations before sampling constellations for $k > 1$. Translations for $k = 1$ can be considered as a subset of transformations of $k = 3$ which should in general fit the data better. On lines 7 and 8 of Algorithm listing 1 translations are sampled and evaluated before full constellation sampling. The routines RANK and APPLY are discussed in Subsection 2.2. While stochastic sub-sampling could be used, for our data sizes it is efficient to simply consider all possible point pairings when $k = 1$.

Constellation sampling Constellation sampling is preferably *seeded* with linkages found by translations so that likely outliers are excluded from consideration. It is worth emphasising that this not only reduces the candidate points to likely inlier points but *constellation congruences* can be identified i.e. the *same set of points* in one frame, can be paired to the *same set of points* in the other frame when proposing transformations. By first finding possible object linkages using translations only, constellation pairs $\mathcal{P}(N_L, 3), \mathcal{P}(M_L, 3)$ can then be sampled from the L inlier objects that appear to be in both frames. These can then be used to generate proposal transformations.



(a) Transformation e.g. taking (blue) 10 to (green) 9 (b) Consensus: displacement mean (c) More accurate consensus transformation

Fig 3. Translation sampling with $k = 1$: Panel (a) shows an arbitrary translation between a sample point in V and a sample point in U. Panel (b) shows the displacement of many such translations from the origin and includes a "consensus" translation i.e. the average of the top ranked translations. Panel (c) illustrates that in this case, the consensus translation is better than the one in panel (a) making better matches between circles and green dots. In this case, while not discernible from the image, the consensus translation turns out to be better than any of the other sampled translations.

2.2 Ranking transformations

Transformation *ranking* is carried out using a modified least squares loss function. The *least squares loss* objective is generally described as the minimization of $\sum |\mathbf{y} - \hat{\mathbf{y}}|^2$, where $\mathbf{y}$ is a *proposal* vector and $\hat{\mathbf{y}}$ is the *target* vector. In the registration problem, the proposal vectors correspond to the points in $U(t)$ under transformations from the set Γ. The target vectors are points in $V(t - \tau)$. The *least squares loss* objective is modified based on the following prior: *(i) we expect a global rigid transformation to explain the movement of object centroids up to small fluctuations due to individual object movement which are deemed negligible. (ii) A distance between objects' centroids* $> \varepsilon$ *is assumed.* The value of ε in the context of the sample data corresponds to a value slightly larger than the average blob's radius. Further details about the role of this parameter in calibrating the tracker are discussed in Subsection 3.2.1. When considering distances between proposal points and target points, only distances to the *first nearest neighbour within a radial distance* ε are considered. Each candidate transformation γ will be applied to *all* points $u_i(t) \in U(t), i \in \{1, 2, ..., N\}$. Let $\tilde{u}_i = \gamma(u_i)$ then we use a cost function

$$C_\gamma = \sum_{i=1}^{N} \min(\varepsilon, |\tilde{u}_i - nn_V(\tilde{u}_i)|^2) \tag{1}$$

where $nn_V(\tilde{U}_i)$ is the nearest neighbour position in $V(t - \tau)$ to the projected point $\tilde{U}_i$. If the projected point has no nearest neighbour within a ball of radius ε, the *capped* distance $\varepsilon + 1$ is attributed. This is illustrated in Fig. 5.

Binning Depending on the data, the cost function C_γ may be sufficient to rank transformations in certain scenarios. Raw transformation scores are real numbers, which are then approximated with integers, binned into discrete classes and ranked using further criteria. Ranking can sort first on the binned score and then sub-sort on the translation displacement magnitude of transformations to distinguish similarly performing transformations. Integer binning is a simple example of a linear binning function that is easy to implement and test. More adaptive approaches to binned ranking such as density-based clustering are potentially more robust but more difficult to test in non-trivial pipelines.

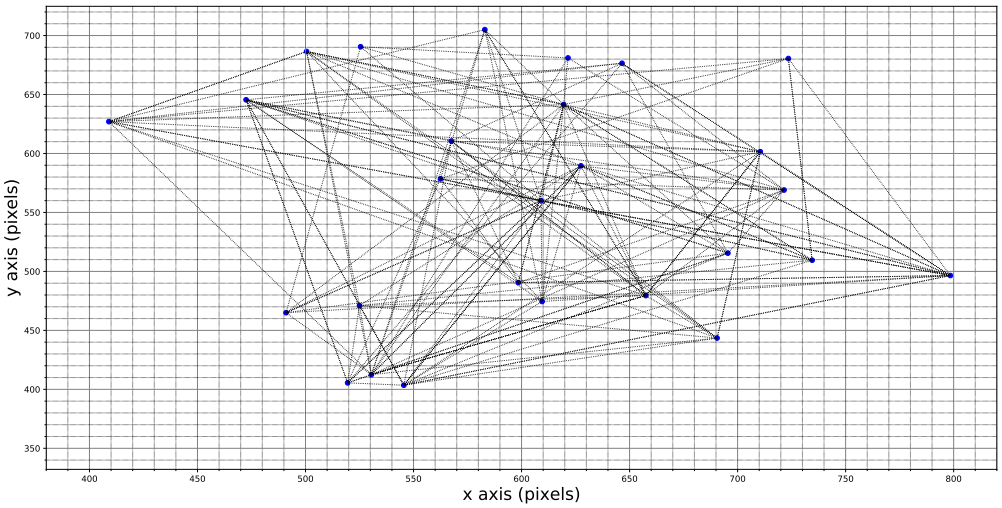


Fig 4. Constellation sampling: Random permutations $\mathcal{P}(M,k)$ with $k=3$ produce triangular constellations in each frame. Affine transformations in 3 dimensions projecting point triplets in one frame to point triplets in another frame will be added to a proposal set Γ for evaluation. While constellations can be sampled randomly and arbitrarily from U and V, seeding with linkages found by translation is more efficient as explained in the main text.

2.3 Some modelling assumptions

A number of assumptions are discussed in this section corresponding to some of the method calls in the algorithm listing. In the PRIORS routine additional transformations beyond the transformations generated from the data can be chosen based on modelling assumptions. Additionally, the filter routines on lines 16-18 can filter data from the final result rather than during the tracking process based on modelling assumptions.

2.3.1 Catchment region

Objects in the point cloud are expected to be separated by a minim distance ε such that under global transformation single objects are matched. The parameter ε can be chosen based on the data. For sparse point cloud data there will typically be only one match within the ball of radius ε. An ε value can be chosen such that it is possible to find multiple objects matched to one object. The interpretation in this case is that all matched points in the catchment region correspond to the same object and that multiple points are the result of *e.g.* image processing anomalies. Merging lineages can be useful to avoid generating superfluous new identifiers. This adds robustness in the event of noisy data or image processing anomalies. Alternatives to using a ball of radius ε consider anisotropic catchment regions that factor in the direction of the global transformation. This is discussed in Section 3.

2.3.2 Consensus translation

Transformations explain the global translation up to small Gaussian fluctuations in individual object movement due to true events or observation error. Fig. 3B gives an example of taking the "consensus" *i.e.* displacement mean of the best translations. This can often be better than any single translation given that each object undergoes a small random fluctuation such that translations between any point-pair cannot explain all

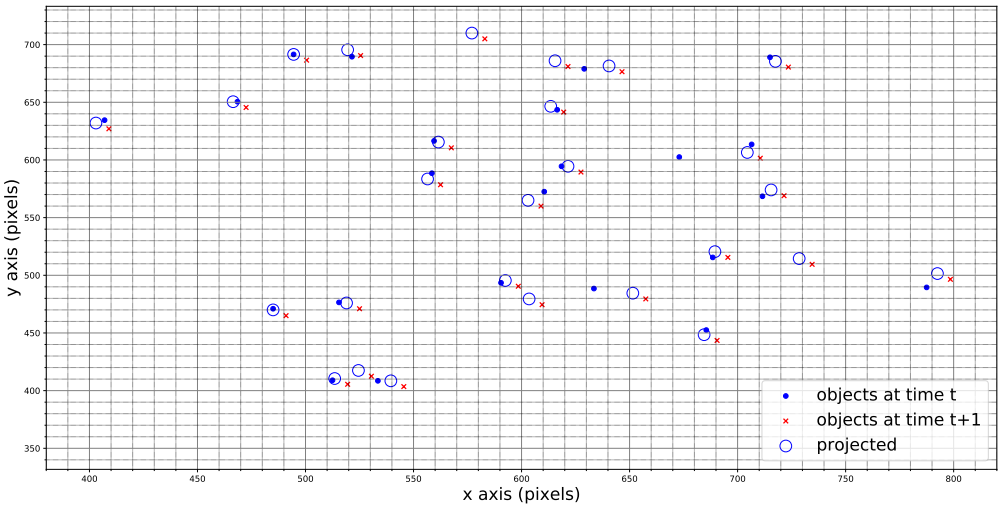


Fig 5. Evaluating transformations: Red crosses correspond to the set $V(t-\tau)$ while the blue dots correspond to objects at the current time $U(t)$. The blue circles with radius ε correspond to the transformation of the set $U(t)$ under the proposal transformation. The transformation score is the average capped distance to nearest neighbours.

object movements. If the position of an object is given as

$$\mathbf{p}_i(t) = \gamma\big(\mathbf{p_i}(\mathbf{t}-\tau\big) + \xi_i$$

where $\xi_i \sim \mathcal{N}(0, \sigma_0)$ is the random fluctuation for one object, taking the "consensus" effectively filters out the *noise* to reveal the global movement.

2.3.3 Minimum allowed frame detections

Transient events are expected to persist for a minimum number of frames greater than or equal to 1. For our data the minimum number of frames is 2. Objects that have been identified for less than 2 frames are excluded from the tracking result. Such object detections are termed *single-frame detections*.

2.3.4 Gaussian region of interest likelihood

Objects of interest in the xy-plane are expected to be found in the center of the field of view. Objects may be detected near the original frame boundaries depending on the size of the ROI. Objects can be treated as inliers/outliers based on standard outlier detection methods under the assumption of normality.

2.4 Software implementation notes

CPU usage is dominated by the image processing stage. For the light-sheet microscopy experiments used in our evaluation, 3D images are approximately 90,000KB on disk with dimensions $(1392, 1040, 60)$, in (x, y, z) order. The software was developed and tested on a personal laptop computer with 16GB RAM and a 2.9-GHz Intel Core i7-3520M CPU. While times vary between different pipeline modes, on average individual frame processing takes around 40 seconds. More than half of the time spent on image processing is spent on de-noising and smoothing images. A negligible fraction of CPU time corresponds to the tracking stage of the pipeline. The tracking stage alone processes 450 data frames in about 1.5 minutes. The current implementation contains significant "meta analysis" overhead.

3 Results and Discussion

3.1 Output

The *life matrix* shown in Fig. 6 is a binary matrix taking a value 1 if an object is detected and 0 otherwise, where each row contains the lifetime of a single detected object, or event. The data in the life matrix are conditioned (filtered) on individuals surviving for at least 2 frames. The tracker requires the frame-frame movement of objects of interest to follow the global transformation. Debris movement is not correlated with that of objects of interest and should not be identified over consecutive frames. These objects appear in the tracking output as single-frame detections and are discarded.

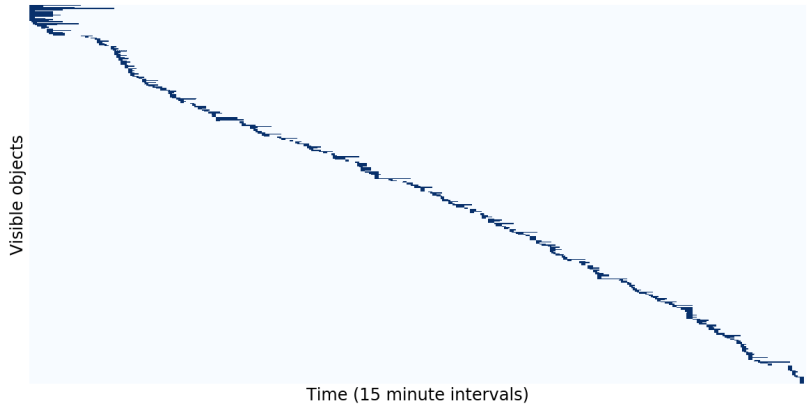


Fig 6. Life matrix: Rows (y-axis) correspond to detected individuals with time on the x-axis. The life matrix shows a generally consistent trend in the appearance and persistence of objects. At the beginning of the sequence, objects persist for an abnormally long time for biological/experimental reasons.

The number of objects observed as a function of time and the average duration of events as a function of time are alternative representations of the data shown in the life matrix. These time series are shown in Fig. 7

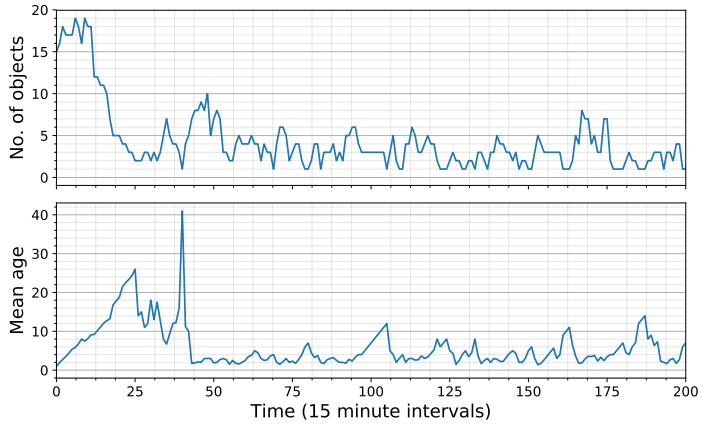


Fig 7. The population time series reflect the information shown in the life matrix. The high spike in the mean age is a result of one object surviving for a long time while others disappear. After this point we observe more typical fluctuations in object activity.

3.2 Output evaluation

In this section we discuss a number of evaluation metrics. Calibration of the characteristic parameter ε at the beginning of an experiment is discussed in 3.2.1. In 3.2.2 we take a closer look at choosing between transformations. In 3.2.3, post-processing is discussed. This stage allows for refinements that can filter out possible false-positives without otherwise affecting the overall tracking result.

3.2.1 Calibration

In our dataset, we use the same value of ε for the image processing stages (e.g. peak detection) and the tracking stage to determine the catchment region. The value of ε in the context of the sample data corresponds to a value slightly larger than the average object radius. The parameter value for the tracker should consider fluctuations around the object size due to processing anomalies or genuine small individual object fluctuations away from the global transformation. In image-space, from inspection of the image data where typical object sizes can be measured, a value of ≈ 20 voxels is expected. The value ε is chosen to minimise the number of single-frame detections where there is ambiguity over the choice of parameters. Fig. 8 illustrates that as we approach this value from below, the number of single-frame detections reduces and then begins to level out. An upper bound on the value of ε, ≈ 20 might be chosen in this case. The goal is not to completely eliminate single-frame detections as indeed objects that persist for only one frame occur either as debris or general image processing anomalies. But in situations where a small change in ε value can increase the number of pairings that the tracker can accurately make, this is preferred. There is a trade-off for $\varepsilon > 20$ where erroneous identity linkages may occur (depending on the characteristic object separation for the data). Some of these linkages may be between objects of interest and debris and some may be debris-debris. In the later case, a large epsilon value allows debris that is visible for multiple frames but moving against the global transformation to be identified over multiple frames. This leads to false positives which should be minimized. A smaller epsilon value will, in contrast, penalise objects that are consistently moving against the global transformation. Values $\varepsilon \ll 20$ lead to failed linkages, generation of new identifiers for outliers and over-counting of objects.

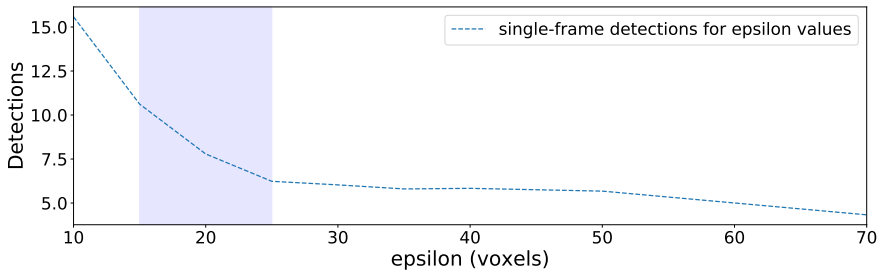


Fig 8. Choosing an ε value: The region 20 ± 5, which is shaded in the plot, shows when the reduction in single-frame detections begins to decrease before levelling out. This region can also be interpreted as the range of observed object radii. There is a trade-off in choosing an ε value. A value of $\varepsilon \ll 20$ results in excessive single frame detections as the tracker cannot accommodate small fluctuations in cell movement because the catchment region is too small, while values for $\varepsilon > 25$ merge lineages erroneously because the catchment region is too large.

3.2.2 Evaluating transformations chosen by tracker

Constellation sampling with $k = 3$ is the default mode for finding a general affine transformation between constellations. This assumes there are at least $k = 3$ objects in each frame and to find a congruent constellation in each frame typically a higher number *e.g.* $2k$ is required. In addition to affine transformations between constellations, translations between single points are always added to transformation proposals. In some cases these may be better than any general affine transformation sampled by the tracker. In situations where there are too few points in either frame to find viable general affine transformations via constellation sampling, the tracker "falls back" to using translations between points as the sampling strategy. Additional transformations can be added to the proposal set using other strategies such as inclusion of an optimal transformation found in the previous time step. Tracking is most accurate when there are many objects in point clouds from which to generate transformations. Fig. 9 shows an example of a sequence with *few* objects. If there are two objects in one frame and one object in the other, translations taking one object to either of the other objects will score similarly. There is no way to clearly distinguish them, keeping in mind that registration and tracking is being performed simultaneously. In the first-to-second frame transition in Fig. 9, the actual object 211 is incorrectly labelled as *new* object 216 and the new object is incorrectly labelled 211. The identity transformation, which was not selected here, scores similarly but not necessarily better than the exhibited rival transformation as both transformations match one object. Binning as discussed in section 2.2 helps to discriminate between similar transformations based on displacement. The transformations will be in the same score "bin" but the identity will be preferred by the tracker because it has the smaller displacement.

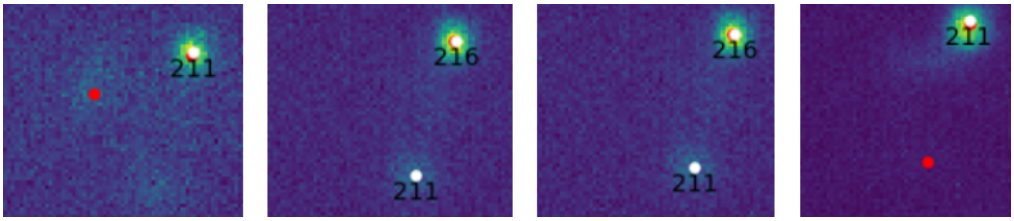


Fig 9. Dealing with low information when ranking transformations: The image sequence (time flows left to right) demonstrates how identifiers can be incorrectly assigned when there are too few objects to generate and rank transformations as explained in the main text. White dots and red dots correspond to objects at time t and $t-1$ respectively. Using the identity transformation *would* make the correct assignment in this example.

3.2.3 Post-processing

Here we discuss analysis and data filtering that can be done in post-processing stages to refine the tracking result.

Anisotropic matching tolerance considers the trade-off when choosing the ε value. When the value is very small, the tracker is very strict, marrying objects that sit within a small ball of radius ε. If the input data are accurate and if a bound on fluctuations away from the global transformation is known and small enough, then a small ε value is sensible. If however there are unpredictable fluctuations for example due to inconsistent centroid/peak detections in noisy data, then a large epsilon value is required to tolerate these fluctuations and to avoid generating new identifiers erroneously. For some datasets it may be difficult to find a value for ε that balances conflicting objectives. One way to mitigate this is to use a larger ε value and then in a post-processing step

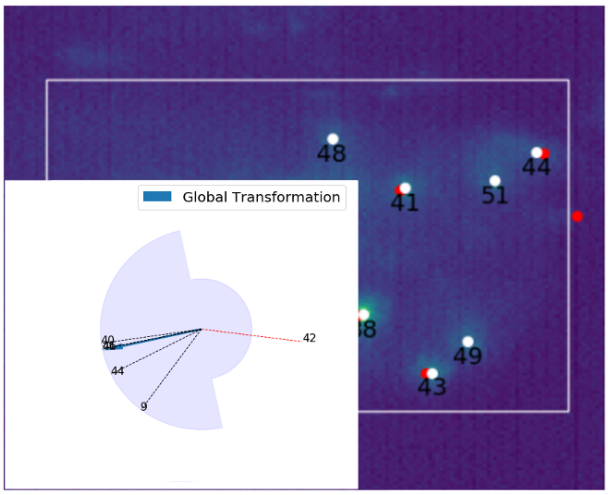
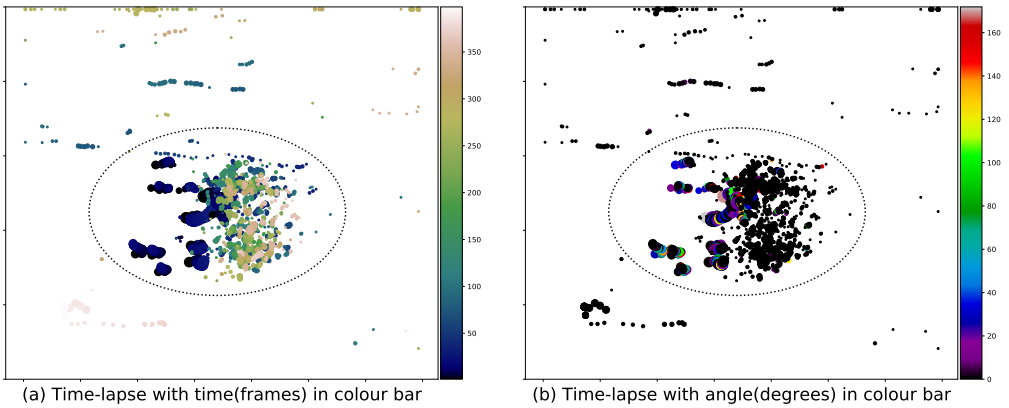


Fig 10. Anisotropic matching tolerance: Differences in angles between the global transformation and the displacements between married objects are compared. White dots and red dots correspond to objects at time t and $t-1$ respectively. Most red dots are to the left of white dots which corresponds to a global transformation of about 170° from the positive x-axis vector. Object 44 has a very different angle to the rest and in the image the red dot appears to the right. Large displacements in conjunction with large angle differences for a number of consecutive frames suggests debris.

replace the isotropic ball of radius ε with an anisotropic catchment region. This penalises objects that are moving in the opposite direction to the global transformation while being more tolerant of objects moving in a similar direction to the global transformation. This is illustrated in Fig. 10 where it is clear object 44 is moving in a different direction to the others. This deviation in movement may or may not be explained as a small object fluctuation.

Data filtering can be applied in a post-processing stage. Fig. 11 illustrates post-processing of the full datasets. The ellipse shows the region containing the most likely mitotic events occurring in the root tip. The time colouring Fig. 11A illustrates when and where events occur and indicates (roughly) if plotted points correspond to the same object. In Fig. 11B, large angle differences between object displacement vectors and a global transformation vector appear in lighter colours (*e.g.* red) and are indicative of possible inlier misclassification. Objects outside of (or far from) the ellipse and objects that show large angle differences are candidates for removal. In the case of angle differences, actual displacements and ε values can be taken into account when filtering objects as described in the preceding paragraph and illustrated in Fig. 10. Fluctuations of the ROI area in each frame is an indicator for how well the image processing step can localise the root. When the signal to noise ratio is good, the root region can be accurately detected and there should be significant fluorescent activity corresponding to biological events within a (relatively) small ROI which varies smoothly between frames. On the other hand if there is too much noise or few (biological) events, it may be difficult to localise the root from frame to frame resulting in a relatively large ROI or one that varies non-smoothly. In the later case, debris may lie within the excessively large ROI and be picked up by the tracker. In Fig. 11, this debris appears for example in the top part of the images, particularly towards the later stages of the experiment due to drift in microscope calibration or other factors. The root grows in the positive x direction (left to right) and debris are typically static or moving with gravity in the opposite direction as seen by some trail-like patterns roughly parallel to the x-axis.



(a) Time-lapse with time(frames) in colour bar (b) Time-lapse with angle(degrees) in colour bar

Fig 11. Time-lapse plots: Two different colourings are used to distinguish time(a) or differences between the object displacement vector and the global transformation vector(b). The ellipse represents the (Gaussian) statistical region of interest over many frames.

Lag analysis can identify possible tracker calibration issues. The tracker considers transformations between consecutive frames, $t-1 \leftarrow t$. Suppose for the purpose of analysis we introduce a lag τ and consider transformations between frames $t-\tau \leftarrow t$. As a confidence indicator, we compare the propagation of object identifiers between frames separated by different lags. Conditioned on objects being more than τ frames old, we compare identifiers propagated over frames for $\tau \in \{1,2,3\}$. We then count the number of objects where identifiers are in agreement between different lags. For example we may expect the following equivalence

$$t-3 \xleftarrow{\lambda_1} t-2 \xleftarrow{\lambda_1} t-1 \xleftarrow{\lambda_1} t \equiv t-3 \xleftarrow{\lambda_3} t \tag{2}$$

where with an abuse of notation λ_τ corresponds to propagation of object identifiers using optimal transformations at lag τ. If the lag analysis shows *significant* differences, it may be due to accumulation of errors, in turn due to poorly chosen parameters. Lag analysis can identity specific instances of violating frames and objects for troubleshooting. The lag analysis result for a sequence of sample frames is shown in Fig. 12 showing generally good agreement between different lags for suitably chosen parameters. While large lag-lag differences suggest calibration issues, small discrepancies

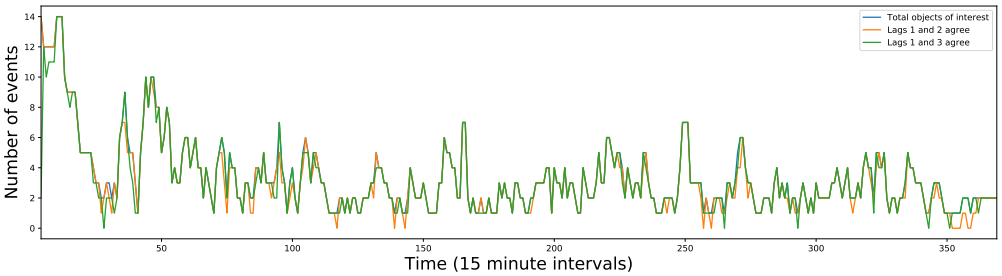


Fig 12. Dis/agreement between lags: Lag-1 identifiers for objects of interest (excluding single-frame detections) are compared with identifiers assigned by $\lambda_\tau : \tau \in \{2,3\}$ transformations for objects that are older than τ. We counted 734 objects in a sample sequence. 42 $\approx 6\%$ showed lag1-2 disagreements and excluding these, 29 $\approx 4\%$ showed lag1-3 disagreements in specific frames.

are expected. For example if there is a sufficient accumulation of individual object fluctuations between lags, which place an object outside of a given catchment region(ε value), an object may be treated as an outlier and given a fresh identifier for larger lags.

3.3 Discussion

In this work we addressed the problem of tracking transient fluorescent events in structured point clouds. We presented a new tracking method in the context of a sample dataset where mitotic events were tracked in *Arabidopsis* roots. Due to the transient nature of fluorescent events and variation in image quality in experiments lasting up to one week, we found a lack of guaranteed, permanent features in image sequences, which greatly influenced the design of the algorithm. The lack of fiducial markers led us to construct fuzzy markers through sub-sampling point clouds. We then constructed suitable cost functions to evaluate affine transformations between these fuzzy fiducial markers. We believe this approach is distinct from registration methods which typically put greater emphasis on the existence of inlier objects.

Furthermore, our method is distinct from existing single particle tracking techniques as we exploit a rigid-structure prior. By using transient fluorescent markers to track mitotic events, we exchanged a morphological analysis problem for a fuzzy registration-tracking problem. The tracking algorithm applies transformations generated from sub-sampling the point cloud data using various strategies and evaluated over the full frame data. The algorithm models frame-frame object movement as a superposition of (a) global affine transformations due to global movement of the structure or the observer and (b) small object fluctuations within the structure.

In extended light-sheet microscopy datasets, image processing and object detection relies heavily on checking and responding to variability in image properties. This goes beyond adaptive thresholding and may result in taking different routes through an image processing pipeline. Isolation of regions of interest within noisy point cloud data is also important.

We considered the role of certain model assumptions in coping with variable and noisy data. (i) Object position updates are treated as small fluctuations away from a global transformation. (ii) The choice of object "separation characteristic" ε is of central importance at all stages of the image processing and tracking pipeline. When considering the tracker in isolation, the ε value plays an additional role: it may *merge* objects which an object detection stage discriminates between if those objects are not sufficiently separated. In the case of accurately counting mitotic events, this reduces the introduction of false-positives *i.e.* the erroneous generation of new identifiers. (iii) Not favouring smaller displacements in transformations has a very negative effect on tracking when the number of objects within frames diminishes. (iv) Object location and angle differences between object displacements and the global transformation may be used in a post-processing stage to remove misclassified outliers.

As useful as these model assumptions are in calibrating the tracker for our data, they also describe situations where the tracker would *not* be suitable. For example if objects are so densely populated that there is no useful minimum separation ε or if objects move independently such that the magnitude of individual movements are of a similar order to (or greater than) the magnitude of the global displacement, the algorithm will be ineffective.

The algorithm as discussed here has not been optimised for large point clouds. The point clouds we worked with contain less than 100 objects in each frame and often less than 20. As the frames are large 3D volumetric images, image processing is the performance bottleneck. Consequently, we have focused on the analysis and flexibility of the tracking algorithm instead of optimisation, at this stage. The transformation sampling and evaluation stages have been implemented efficiently within a vector-programming paradigm. Focus on performance improvements should emphasise efficiency of the *transformation proposal sampling*. This should, in general, be considered *NP-hard* as proposals are generated by finding congruences in large point cloud data. For point clouds with high objects counts, proposal sampling could be

applied on a suitable subregion of the data to avoid excessive evaluations. 443

Our approach was motivated by the case of transient florescent markers used in biology, where sub-cellular fluctuations are superposed with large-scale movements due to tissue growth. We expect that our approach will be applicable in other situations where tracking transient events embedded in rigid structures is required. A possible but more complex extension could relax the constraint of a single global affine transformation to an approximate hierarchy of local, structurally-constrained regions within point clouds. Tracking individuals in a swarm of fireflies observed at night comes to mind.

References

1. von Wangenheim D, Hauschild R, Fendrych M, Barone V, Benkova E, Friml J. Live tracking of moving samples in confocal microscopy for vertically grown roots. eLife. 2017;6.

2. Ulman V, Maška M, Magnusson KE, Ronneberger O, Haubold C, Harder N, et al. An objective comparison of cell-tracking algorithms. Nature methods. 2017;14(12):1141.

3. Kanade T, Yin Z, Bise R, Huh S, Eom S, Sandbothe MF, et al. Cell image analysis: Algorithms, system and applications. In: Applications of Computer Vision (WACV), 2011 IEEE Workshop on. IEEE; 2011. p. 374–381.

4. Wöll D, Kölbl C, Stempfle B, Karrenbauer A. A novel method for automatic single molecule tracking of blinking molecules at low intensities. Physical Chemistry Chemical Physics. 2013;15(17):6196–6205.

5. Godinez WJ, Rohr K. Tracking multiple particles in fluorescence time-lapse microscopy images via probabilistic data association. IEEE Trans Med Imaging. 2015;34(2):415–432.

6. Liu AA, Lu Y, Chen M, Su YT. Mitosis Detection in Phase Contrast Microscopy Image Sequences of Stem Cell Populations: A Critical Review. IEEE Transactions on Big Data. 2017;3(4):443–457.

7. Besl PJ, McKay ND. Method for registration of 3-D shapes. In: Sensor Fusion IV: Control Paradigms and Data Structures. vol. 1611. International Society for Optics and Photonics; 1992. p. 586–607.

8. Fitzgibbon AW. Robust registration of 2D and 3D point sets. Image and vision computing. 2003;21(13-14):1145–1153.

9. Jian B, Vemuri BC. Robust point set registration using gaussian mixture models. IEEE transactions on pattern analysis and machine intelligence. 2011;33(8):1633–1645.

10. Zhou Z, Tu J, Geng C, Hu J, Tong B, Ji J, et al. Accurate and Robust Non-rigid Point Set Registration using Student'st Mixture Model with Prior Probability Modeling. Scientific reports. 2018;8(1):8742.

11. Reid D, et al. An algorithm for tracking multiple targets. IEEE transactions on Automatic Control. 1979;24(6):843–854.

12. Cox IJ, Hingorani SL. An efficient implementation of Reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking. IEEE Transactions on pattern analysis and machine intelligence. 1996;18(2):138–150.

13. Schulz D, Burgard W, Fox D, Cremers AB. Tracking multiple moving targets with a mobile robot using particle filters and statistical data association. In: Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on. vol. 2. IEEE; 2001. p. 1665–1670.

14. Okuma K, Taleghani A, De Freitas N, Little JJ, Lowe DG. A boosted particle filter: Multitarget detection and tracking. In: European conference on computer vision. Springer; 2004. p. 28–39.

15. Chenouard N, Smal I, De Chaumont F, Maška M, Sbalzarini IF, Gong Y, et al. Objective comparison of particle tracking methods. Nature methods. 2014;11(3):281.

16. Amarteifio S, Fallesen T, Sena G, Pruessner G. lightroot; 2018. `https://github.com/GiovanniSena/LSFM_CYCB_analysis_v2`.

17. Sena G, Frentz Z, Birnbaum KD, Leibler S. Quantitation of cellular dynamics in growing Arabidopsis roots with light sheet microscopy. PLoS One. 2011;6(6):e21303.

18. Baesso P, Randall RS, Sena G. Light Sheet Fluorescence Microscopy Optimized for Long-Term Imaging of Arabidopsis Root Development. In: Root Development. Springer; 2018. p. 145–163.

19. Reddy GV, Heisler MG, Ehrhardt DW, Meyerowitz EM. Real-time lineage analysis reveals oriented cell divisions associated with morphogenesis at the shoot apex of Arabidopsis thaliana. Development. 2004;131(17):4225–4237.

20. Donoho DL, Johnstone JM. Ideal spatial adaptation by wavelet shrinkage. biometrika. 1994;81(3):425–455.

21. van der Walt S, Schönberger JL, Nunez-Iglesias J, Boulogne F, Warner JD, Yager N, et al. scikit-image: image processing in Python. PeerJ. 2014;2:e453.

22. Beucher S. Watershed, hierarchical segmentation and waterfall algorithm. In: Mathematical morphology and its applications to image processing. Springer; 1994. p. 69–76.

23. Fischler MA, Bolles RC. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM. 1981;24(6):381–395.