Tracking by weakly-supervised learning and graph optimization for whole-embryo *C. elegans* lineages

Peter Hirsch[1,2] PETER.HIRSCH@MDC-BERLIN.DE
Caroline Malin-Mayor[3] MALINMAYORC@JANELIA.HHMI.ORG
Stephan Preibisch[3] PREIBISCHS@JANELIA.HHMI.ORG
Dagmar Kainmueller[*1] DAGMAR.KAINMUELLER@MDC-BERLIN.DE
Jan Funke[*3] FUNKEJ@JANELIA.HHMI.ORG
[1] *Max-Delbrueck-Center for Molecular Medicine in the Helmholtz Association, DE*
[2] *Humboldt-Universitt zu Berlin, Faculty of Mathematics and Natural Sciences, DE*
[3] *HHMI Janelia Research Campus, USA*

Editors: Under Review for MIDL 2022

Abstract

The tracking of all nuclei of an embryo in noisy and dense fluorescence microscopy data is a challenging task. We build upon a recent method that combines deep learning to extract candidate solutions with an integer linear program (ILP) to select the most likely tracks. We present extensions of this method to specifically address the following challenging properties of *C. elegans* embryo recordings: (1) Relatively many cell divisions compared to benchmark recordings of other organisms, and (2) the presence of polar bodies, which look similar to cell nuclei and are thus easily mistaken as such. To cope with (1), we devise and incorporate a learnt cell division detector. To cope with (2), we devise and incorporate a learnt polar body detector. We further extend the method to allow for automated ILP hyperparameter tuning via a structured SVM, thus alleviating the need for tedious manual set-up of a respective grid search.

At the time of submission, our method heads the leaderboard of the cell tracking challenge on the *Fluo-N3DH-CE C. elegans* embryo dataset. Furthermore, we report an extensive quantitative evaluation of our method on two additional *C. elegans* datasets, namely a set of 3 fully annotated confocal embryo recordings, and a set of 3 fully annotated lightsheet embryo recordings. We will make these datasets public to serve as an extended benchmark for future method development. To gauge the practical impact of our method, we include the software Starrynite as baseline. Starrynite is commonly employed by biologists for the study of *C. elegans*. Our results suggest considerable improvements, especially in terms of the correctness of division event detection and the number of fully correct tracks.

Keywords: detection, tracking, deep learning, ILP, optimization, *C. elegans*, microscopy

1. Introduction

Advances in microscopy have made the recording of whole embryo development possible, even for relatively large model organisms such as zebrafish and mouse (Keller et al., 2010; Krzic et al., 2012). However there is an inherent tradeoff between frame rate, resolution and the prevention of phototoxicity (Weigert et al., 2018). Hence, while it is possible to capture high signal-to-noise images with high resolution, this damages the organism quickly, especially during early embryonic

[*] shared last

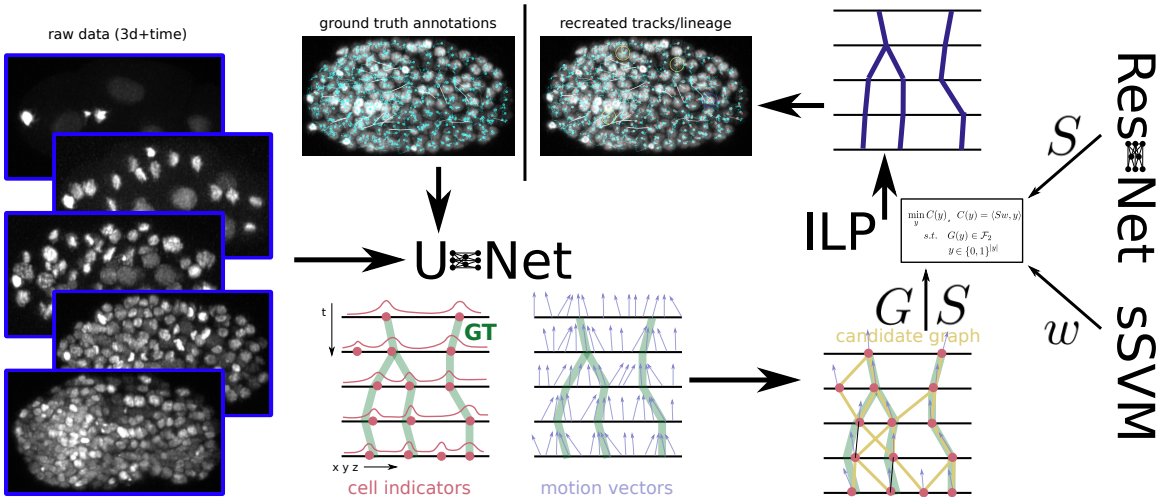

Figure 1: Overview of the method: The network learns to extract cell candidates and motion vectors from the raw data that are used to construct a candidate graph. From this graph, using learned cell state scores and hyperparamters found via a structured SVM, an ILP extracts the lineage.

development. Longer periods of embryonic development can be captured at lower resolution as well as reduced signal-to-noise ratio (SNR). However, low SNR exacerbates the detection of cells. This adds to the challenge posed by fluctuating signal strength in different cell development phases. Furthermore, the low frame rate renders overlap-based tracking approaches inadequate, the low textural variety between nuclei similarity-based ones. While purely manual tracking is theoretically possible for single samples of relatively small organisms, it neither scale to larger organisms nor to larger sets of samples.

To this end, a number of automated cell tracking approaches have been developed that are designed to cope with reduced SNR as well as frame rates on the order of minutes. Such methods have enabled a range of studies on a variety of organisms, where it wouldn't have been feasible to do tracking manually (Li et al., 2019; Murray et al., 2008; Cao et al., 2020; de Medeiros et al., 2021; Wolff et al., 2018; Guignard et al., 2020). The *Cell Tracking Challenge* (Ulman et al., 2017), an extensive benchmark that contains 2d+time and 3d+time datasets of different organisms recorded with a variety of microscopes, allows for a quantitative comparison of automated cell tracking methods.

In our work, we build upon a recent method that combines learning and optimization in a two-step process (Malin-Mayor et al., 2021). We propose extensions of this method to capture properties specific to recordings of the model organism *C. elegans*, namely relatively many cell divisions, and the presence of *polar bodies* which look similar to cell nuclei. These extensions yield significantly reduced errors on said data. Furthermore, we propose an extension that allows for ease of hyperparameter tuning, namely alleviating the need for manual configuration of a grid search by means of a structured SVM. In summary, our contributions are:

- A learnt cell state and polar body detector, integrated into an existing approach that combines deep learning and an integer linear program (ILP) for nuclei tracking.
- Fully automated tuning of the hyperparameters of the ILP via structured SVM.

- Our method defines the new state-of-the-art for *C. elegans* cell tracking in the Cell Tracking Challenge.
- Our method improves significantly over the tool currently used by practitioners on this type of data, Starrynite, thereby reducing the time required for manual curation.
- We make a new dataset of three fully annotated confocal recordings and three fully annotated lightsheet recordings of *C. elegans* available as a benchmark for future methods.

2. Related Work

Cell tracking methods can be broadly divided into two categories, namely *tracking by model evolution* and *tracking by detection/assignment*.

Tracking-by-detection methods first compute (candidate) cell detections in all frames, and in a second step link matching cell detections across frames. E.g., Starrynite (Bao et al., 2006) uses classical computer vision to detect locations of maximum signal in each frame, nearest neighbor matching for the linking detections, and local post-processing to resolve ambiguities that occur in case of cell divisions. More recently, Cao et al. (2020) replaced the classical detection step of Starrynite by neural network-based cell segmentation. Detections can also be linked in a globally optimal manner by means of combinatorial optimization, for a set of cell detections that is assumed to be correct (Magnusson et al., 2015), as well as in the face of over- and underdetections that may thus be amended (Schiegg et al., 2013), as well as for an overcomplete set of candidate detections from which a feasible subset is thus extracted (e.g., Jug et al., 2014).

Tracking-by-model-evolution methods first detect cells in a key frame (usually the first or last), constructs a model based on these, and then evolves the model by iteratively fitting it to the data in the next frame, constrained by the previous frame (e.g., the number of cells). One example of this category that is capable of handling very large datasets is TGMM (Amat et al., 2014; McDole et al., 2018) that uses Gaussian Mixture Models. Other examples use active meshes, contours or level sets (Dufour et al., 2010; Sun et al., 2020; Ray and Acton, 2002).

An important aspect of any tracking model of living cells is to observe certain biological constraints. Cells are limited in the way they evolve – e.g., a cell can divide into two (but not more) cells, and two cells will never merge into one over time. Tracking-by-model-evolution methods typically construct a respective feasible tracking solution step-by-step, while tracking-by-assignment approaches employ respective feasibility constraints as part of some optimization method (Kausler et al., 2012; Schiegg et al., 2013; Jug et al., 2014; Schiegg et al., 2015; Jug et al., 2016; Haubold et al., 2016).

3. Method

Our method extends the tracking-by-detection approach of Malin-Mayor et al. (2021). For an overview of the (extended) method see Figure 1. Their detection step employs a deep neural network to predict the position of each nucleus and its position in the previous time frame via a motion vector. To cope with strong cell movement, for example during cell division and in later frames, they employ a global optimization procedure for linking based on integer linear programming (ILP) to exploit temporal context and incorporate prior knowledge, such as the fact that a cell can only divide into two but not more daughter cells.

In our work, we propose to include a separate network to classify the cell state to further improve the performance. As dividing cells are relatively rare in comparison to non-dividing cells, this is intended to help the ILP to correctly link cells, especially in denser areas. Furthermore, we propose the use of a structured SVM (sSVM) to automatically find the optimal hyperparamters for the ILP.

By default, we do not perform any postprocessing on the tracks (such as removal of short tracklets). However, we propose one respective exception, namely concerning the *polar bodies*, a peculiarity of embryonic development that appears especially in *C. elegans*. A polar body is a cell that is formed at the same time as the egg cell but cannot be fertilized and does not divide any further (see Suppl. Figure 2 for an example). They do play an important role in the early development of *C. elegans*. However, depending on the aspect studied they might not be of interest, and even if, they are not contained in the ground truth tracks as they are not considered "proper" cells. That is why we add them as an additional class to our cell state classifier and can thus optionally remove them from the tracks. See Suppl. Sec. A.8 for details on how they are removed.

3.1. Deep Learning based Prediction

We use a 4d U-Net (Ronneberger et al., 2015; Cicek et al., 2016) to detect potential nuclei candidates and their motion vector. We use a 3d ResNet (He et al., 2015) as cell state classifier. For more details on the architecture and the training and inference procedures see Suppl. Sec. A.2.

3.1.1. CELL CANDIDATES

To detect nuclei we follow the approach of Malin-Mayor et al. (2021) and Höfener et al. (2018): A Gaussian-shaped blob is placed at the location of every ground truth annotation and regressed. During inference we employ a max pooling layer with stride one and a window size slightly smaller than the size of a nucleus to perform non-maximum suppression (NMS) and extract the maxima to serve as our cell candidates.

3.1.2. MOTION VECTORS

Additionally we learn to predict the motion of a cell between adjacent time frames. Following Malin-Mayor et al. (2021), and similarly to Hayashida et al. (2020), we predict a vector pointing backwards in time to the position in the previous time frame. A characteristic of developing embryos is that objects, the cells, can only split going forward but not merge. Therefore every cell, if there are no field of view of the microscope related issues and with the exception of the first frame, has exactly one predecessor, but might have zero to two successors, zero in case of apoptosis (cell death). This simplifies tracking backwards.

During inference we extract the motion vectors corresponding to our cell candidates from the first network.

3.1.3. CELL STATE

We propose to incorporate a classifier to determine the cell state of each detection similarly to Santella et al. (2014). We assign to each detection one of four classes: parent cell (cell that is about to undergo cell division), daughter cell (cell that just divided), polar body and none of those (continuation). We use a ResNet-18 with 3d convolutions for this classification task. The scores for the parent/daughter/continuation classes are incorporated as costs into the ILP (see Sec. 3.2). The score for the polar body class is used in the optional postprocessing (see Suppl. Sec. A.8).

3.2. ILP-based Linking

3.2.1. GRAPH-BASED OPTIMIZATION

Following Malin-Mayor et al. (2021), we use integer linear programming based optimization to extract and improve the final tracks. The neural network predictions are used to construct a candidate graph $G = (V, E)$. All detections v with position p_v at time frame t_v after the NMS with a score s_v above some threshold th (we used 0.2 in all cases) are nodes V in the graph. To construct the edges E we use the motion vectors m_v. For each node v we compute its predicted position $\hat{p}_v = p_v + m_v$ in the previous time frame $t_v - 1$. We collect every node u in $t_v - 1$ that is within some distance d of that position and connect it with an edge $e = (v, u)$ to the node v.

The goal is to extract the binary forest from this graph with the minimal cost according to the following objective and constraints, in formal terms:

$$\min_y C(y) \quad s.t. \quad G(y) \in \mathcal{F}_2 \tag{1}$$

$\mathcal{F}_2$ refers to the set of all possible binary forests, y to a valid set of selected nodes and edges represented by an indicator vector and $C\colon y \to \mathbb{R}$ maps a cost to each such set. A set is valid if it adheres to the given constraints. To compute the cost function we construct a sparse feature matrix $S^{dim(y)\times dim(w)}$ based on the learnt tracking and cell state scores with one row per indicator. We collect all ILP hyperparameters in a tunable weight vector w. Given some w the cost to minimize is then

$$C(y) = \langle Sw, y \rangle. \tag{2}$$

See Suppl. Sec. A.3 for more information on how the vector is constructed and how S and w are defined.

In a last step we add a set of constraints. A valid solution has to be (biologically) morally sound (e.g. cells can only divide into two daughter cells) and consistent (e.g. if an edge is selected, its endpoint nodes have to be selected as well). For an overview of all constraints see Suppl. Sec. A.4.

We use Gurobi (Gurobi Optimization, 2021) for block-wise solving (see Suppl. Sec. A.3)

3.2.2. STRUCTURED SVM-BASED HYPERPARAMETERS SEARCH

The initial approach to find a good parameter vector w for the problem above is to perform a grid search within some predefined range. However, if that range is unknown, this can be costly and a new, matching range has potentially to be found for each new type of data. Given the solution y' a modified objective can also be solved for optimal weights w', the ILP hyperparameters, instead of optimal tracks. This can be done using a structured SVM (sSVM) with the objective

$$\min_w \left(\lambda |w|^2 + L(w)\right), \tag{3}$$

with L being the soft margin loss

$$L(w) = \max_y \left(\langle w, S^T y' - S^T y \rangle + \Delta(y', y)\right) \tag{4}$$

with a Hamming cost function Δ and a regularization factor λ.

With the help of the ground truth annotations we can compute a "best effort" indicator vector to be used as y'. This equates to the best possible solution given the set of predicted cell candidates and

motion vectors. The same constraints on y as in the original problem hold. The set of constraints, the best effort indicator vector y' and the feature matrix S are passed to a solver (Funke, 2017) that returns the weight vector w'.

4. Results

To measure the performance of our method we evaluate it on three different datasets of developing *C. elegans* embryos, the **Fluo-N3DH-CE** dataset of the Cell Tracking Challenge benchmark (CTC, Ulman et al., 2017), three confocal recordings (**mskcc_confocal**) and three lightsheet recordings (**nih_ls**). See Suppl. Sec. A.1 for more details on the data. Our method performs well on all three, and heads, at the time of submission, the leaderboard for the tracking (TRA) and detection (DET) scores of the CTC for this dataset.

4.1. CTC *C. elegans* data

The **Fluo-N3DH-CE** dataset (Murray et al., 2008) consists of four 3d+time anisotropic confocal recordings approx. until the 350 cell stage; 2 public ones for training and 2 private ones for the official evaluation. All tracks are annotated. At the time of submission our method heads the leaderboard for this dataset for the detection score (DET) and tracking score (TRA) out of 14 submissions, outperforming the previous state of the art from Sugawara et al. (2021) (see Table 1).[1]

Table 1: Quantitative results for the Fluo-N3DH-CE *C. elegans* embyro test dataset of the Cell Tracking Challgene. The DET score measures detection performance, the TRA score tracking performance according to Matula et al. (2015) with 1 being the perfect score.

Fluo-N3DH-CE	DET	TRA
ours	**0.981**	**0.979**
Elephant (Sugawara et al., 2021)	0.979	0.975
Baxter (Magnusson et al., 2015)	0.959	0.945

4.2. Confocal *C. elegans* data

The **mskcc_confocal** dataset consists of three longer fully annotated 3d+time anisotropic confocal recordings (Santella et al., 2014). The ground truth has been created using Starrynite (Bao et al., 2006; Santella et al., 2014), followed by manual curation. We use the uncurated Starrynite results as a baseline. We train and evaluate all our models on the first 270 frames. Most divisions of the *C. elegans* lineage have already occurred by the 270th frame and some cells have already undergone apoptosis. In addition we evaluate the same models on the first 200 frames, which is in line with many studies conducted on *C. elegans* (see Suppl. Sec. A.9).

For each experimental run we use one recording as training data, one for validation and one as our test set. We do this for all six possible combinations. For each combination, we perform three experimental runs, starting from different (standard random) weight initializations, leading to

1. The online leaderboards for the Cell Tracking Challenge can be found at http://celltrackingchallenge.net/latest-ctb-results (TRA) and http://celltrackingchallenge.net/latest-csb-results (DET). Note that our results (named JAN-US) have not been made available there yet at the time of submission.

a total of 18 experimental runs. See Table 2 and Suppl. Figure 3 for quantitative results, where each number we report there is obtained by averaging over the 18 runs. Divisions that are off by one frame compared to the annotations are not counted as errors as the limited frame rate leads to inherent inaccuracies in the data and annotations. For comparison with other methods we also report node and edge recall. However their informative value is limited as the values are quite saturated, still, there is some improvement (see Suppl. Table 6).

We conducted an ablation study on the **mskcc_confocal** dataset, measuring the effect of the individual parts of our method (see Table 3 and Suppl. Table 5). Instead of the ILP we perform greedy nearest neighbor matching (while still observing biological correctness). Moreover, we repeat the experiments without incorporating the cell state classifier in the ILP (this matches the system in the prior work (Malin-Mayor et al., 2021)). Both strongly suffer from false positive (fp)-type errors. We add filtering of polar bodies in the postprocessing (both from the predictions and the ground truth). This drastically decreases fps: In addition to removing polar bodies, some other fp detections that can be attributed to noise are also removed. Finally we compare the results with automatically found ILP hyperparameters and with manually configured grid-searched ones, and find that sSVM-determined hyperparameters yield competitive results. The sSVM finds similar hyperparamters for all experimental runs (see Suppl. Figure 4).

Discussion. We did not expect to see large differences between sSVM-determined hyperparameters and manually configured grid search as we have gathered experience in choosing appropriate parameters for the hyperparameter grid search for this data. Thus the explicit search is often faster as it can be parallelized indefinitely. However, for other data, where this information is not at hand, the targeted sSVM is very convenient and is computationally more efficient. Interestingly, depending on the hyperparameters the system appears to be able to exchange fp and fn errors. The sSVM-determined hyperparameters seem to prioritize fp errors. By adapting the cost function Δ one should be able to modulate this depending on respective application-specific needs.

4.3. Lightsheet *C. elegans* data

The **nih_ls** dataset consists of three fully annotated 3d+time isotropic lightsheet recordings (Moyle et al., 2021). The experimental setup is similar to the one for mskcc_confocal. On this data, Starrynite produced large numbers of false positives, thus not delivering a meaningful baseline. We thus merely present our results (see Table 2 and Suppl. Table 4), in combination with making the data publicly available, as a baseline for future method developments.

Discussion. It is interesting to compare our results on **nih_ls** and **mskcc_confocal**: Due to the isotropic resolution of **nih_ls** we expected the results to be superior, yet so far the error metrics we observe do not support this intuition. A closer look at qualitative results reveals some clues that may explain part of it: Apoptotic cells are more distinct and visible earlier in **nih_ls** (see Suppl. Figure 2 for an example) and thus have not been annotated in the ground truth. Yet in the current state our model does not handle this transition explicitly and thus continues to track them temporarily, leading to a larger number of false positives, as indicated by the quantitative results. As we already have a cell state classifier as part of our model, it will be straightforward to add apoptotic cells as a remedy.

5. Conclusion

In this work we presented extensions to (Malin-Mayor et al., 2021) to improve tracking of all cells during embryonic development of *C. elegans*. In addition to combining deep learning to learn

Table 2: Quantitative results for **mskcc_confocal** and **nih_ls**, *FP*: false positive edge, *FN*: false negative edge, *IS*: identity switch/cross-over of tracks, *FP-div*: false positive division, *FN-div*: false negative division, *total div*: sum of division errors, *total*: sum of all errors, normalized per 1k edges. *REFT*: ratio error free tracks (number of cells in the last frame whose reverse track has no error divided by the number of GT cells in the last frame, biased in favor of fp as superfluous tracklets ending earlier are not counted).

	FP	FN	IS	FP div	FN div	total div	total	REFT
mskcc_confocal 270 frames								
Starrynite	7.902	13.321	0.618	0.579	1.184	1.763	23.604	0.769
ours wo/cls	4.757	5.464	0.055	1.122	0.261	1.382	11.659	0.891
ours ssvm	3.684	5.703	0.048	0.066	0.513	0.579	9.909	0.839
nih_ls 270 frames								
ours w/cls	12.047	7.193	0.446	0.317	0.351	0.668	20.353	0.852

Table 3: Ablation study, on **mskcc_confocal** data on 270 frames. We ablate solving an ILP altogether (ILP), incorporating the cell state classifier (cls), employing an sSVM for hyperparameter search (ssvm), and incorporating the polar body filter (pbf). Description of error types see Table 2.

ILP	cls	ssvm	pbf	FP	FN	IS	FP div	FN div	total div	total	REFT
✗	✗	✗	✗	5.008	4.639	0.048	1.600	0.255	1.855	11.550	0.912
✓	✗	✗	✗	4.757	5.464	0.055	1.122	0.261	1.382	11.659	0.891
✓	✓	✗	✗	3.408	5.669	0.028	0.202	0.263	0.464	9.570	0.871
✓	✓	✓	✗	3.684	5.703	0.048	0.066	0.513	0.579	9.909	0.839
✓	✓	✓	✓	2.556	5.717	0.050	0.062	0.510	0.573	8.907	0.836

position and motion vectors of each cell and integer linear programming to extract tracks over time and ensure long term consistency, we integrate cell state information into the ILP, together with a method to automatically determine the ILP hyperparameters, alleviating the need for potentially suboptimal manually configured grid-search.

At the time of submission our method heads the leaderboard of the CTC for the DET and TRA scores for the **Fluo-N3DH-CE** dataset. On two other datasets of both confocal and lightsheet recordings of *C. elegans* our method outperforms the tool Starrynite, which is often used by practitioners for studies of *C. elegans*, by a wide margin. The low error rate, especially up to the 350 cell stage, will push down the required time for manual curation significantly. This will facilitate studies that require a large number of samples. More effort is still necessary in the later stages of development. In future work we will extend the tracking all the way to the end of the embryonic development. This poses additional challenges as the whole embryo starts to twitch, causing abrupt movements. A second avenue of future work is to combine the two stages of the method. Recent work (Pogani et al., 2020) has proposed a method to incorporate black box solvers into a gradient-based end-to-end neural network learning process. This shows great promise to increase the performance of our method even further.

Acknowledgments

We would like to thank Anthony Santella and Zhirong Bao et al. and Ryan Christensen and Hari Shroff et al. for providing us with their data and annotations, for generously allowing us to make the data public and for a lot of very valuable information and feedback.

References

Fernando Amat, William Lemon, Daniel P Mossing, Katie McDole, Yinan Wan, Kristin Branson, Eugene W Myers, and Philipp J Keller. Fast, accurate reconstruction of cell lineages from large-scale fluorescence microscopy data. *Nature Methods*, 11(9):951–958, 2014. doi: 10.1038/nmeth.3036. URL https://doi.org/10.1038/nmeth.3036.

Z. Bao, J. I. Murray, T. Boyle, S. L. Ooi, M. J. Sandel, and R. H. Waterston. Automated cell lineage tracing in caenorhabditis elegans. *Proceedings of the National Academy of Sciences*, 103(8):2707–2712, 2006. doi: 10.1073/pnas.0511111103. URL https://doi.org/10.1073/pnas.0511111103.

Jianfeng Cao, Guoye Guan, Vincy Wing Sze Ho, Ming-Kin Wong, Lu-Yan Chan, Chao Tang, Zhongying Zhao, and Hong Yan. Establishment of a morphological atlas of the caenorhabditis elegans embryo using deep-learning-based 4d segmentation. *Nature communications*, 11(1): 1–14, 2020.

Ozgun Cicek, Ahmed Abdulkadir, Soeren S. Lienkamp, Thomas Brox, and Olaf Ronneberger. 3d u-net: Learning dense volumetric segmentation from sparse annotation. *CoRR*, 2016. URL http://arxiv.org/abs/1606.06650v1.

Gustavo de Medeiros, Raphael Ortiz, Petr Strnad, Andrea Boni, Francisca Maurer, and Prisca Liberali. Multiscale light-sheet organoid imaging framework. *bioRxiv*, 2021. doi: 10.1101/2021.05.12.443427. URL https://www.biorxiv.org/content/early/2021/05/12/2021.05.12.443427.

Alexandre Dufour, Roman Thibeaux, Elisabeth Labruyere, Nancy Guillen, and Jean-Christophe Olivo-Marin. 3-d active meshes: fast discrete deformable models for cell tracking in 3-d time-lapse microscopy. *IEEE transactions on image processing*, 20(7):1925–1937, 2010.

Jan Funke. *Automatic Neuron Reconstruction from Anisotropic Electron Microscopy Volumes*. PhD thesis, ETH Zrich, 2017.

Léo Guignard, Ulla-Maj Fiúza, Bruno Leggio, Julien Laussu, Emmanuel Faure, Gaël Michelin, Kilian Biasuz, Lars Hufnagel, Grégoire Malandain, Christophe Godin, and Patrick Lemaire. Contact area–dependent cell communication and the morphological invariance of ascidian embryogenesis. *Science*, 369(6500), 2020. ISSN 0036-8075. doi: 10.1126/science.aar5663. URL https://science.sciencemag.org/content/369/6500/eaar5663.

Gurobi Optimization. Gurobi Optimizer Reference Manual, 2021. URL https://www.gurobi.com.

Carsten Haubold, Janez Aleš, Steffen Wolf, and Fred A. Hamprecht. A generalized successive shortest paths solver for tracking dividing targets. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 566–582, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46478-7.

Junya Hayashida, Kazuya Nishimura, and Ryoma Bise. MPM: Joint representation of motion and position map for cell tracking. *CoRR*, 2020. URL http://arxiv.org/abs/2002.10749v2.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, 2015. URL http://arxiv.org/abs/1512.03385v1.

Larissa Heinrich, Jan Funke, Constantin Pape, Juan Nunez-Iglesias, and Stephan Saalfeld. Synaptic cleft segmentation in non-isotropic volume electron microscopy of the complete drosophila brain, 2018.

Henning Höfener, André Homeyer, Nick Weiss, Jesper Molin, Claes F Lundström, and Horst K Hahn. Deep learning nuclei detection: A simple approach can deliver state-of-the-art results. *Computerized Medical Imaging and Graphics*, 70:43–52, 2018.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *CoRR*, 2018. URL http://arxiv.org/abs/1803.05407v3.

Florian Jug, Tobias Pietzsch, Dagmar Kainmüller, Jan Funke, Matthias Kaiser, Erik van Nimwegen, Carsten Rother, and Gene Myers. Optimal joint segmentation and tracking of escherichia coli in the mother machine. In *Bayesian and grAphical Models for Biomedical Imaging*, pages 25–36. Springer, 2014.

Florian Jug, Evgeny Levinkov, Corinna Blasse, Eugene W. Myers, and Bjoern Andres. Moral lineage tracing. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2016. doi: 10.1109/cvpr.2016.638. URL http://dx.doi.org/10.1109/CVPR.2016.638.

Bernhard X Kausler, Martin Schiegg, Bjoern Andres, Martin Lindner, Ullrich Koethe, Heike Leitte, Jochen Wittbrodt, Lars Hufnagel, and Fred A Hamprecht. A discrete chain graph model for 3d+ t cell tracking with high misdetection robustness. In *European Conference on Computer Vision*, pages 144–157. Springer, 2012.

Philipp J Keller, Annette D Schmidt, Anthony Santella, Khaled Khairy, Zhirong Bao, Joachim Wittbrodt, and Ernst HK Stelzer. Fast, high-contrast imaging of animal development with scanned light sheet–based structured-illumination microscopy. *Nature methods*, 7(8):637–642, 2010.

Uros Krzic, Stefan Gunther, Timothy E Saunders, Sebastian J Streichan, and Lars Hufnagel. Multi-view light-sheet microscope for rapid in toto imaging. *Nature methods*, 9(7):730–733, 2012.

Abhishek Kumar, Yicong Wu, Ryan Christensen, Panagiotis Chandris, William Gandler, Evan McCreedy, Alexandra Bokinsky, Daniel A. Colón-Ramos, Zhirong Bao, Matthew McAuliffe, Gary Rondeau, and Hari Shroff. Dual-view plane illumination microscopy for rapid and spatially isotropic imaging. *Nature Protocols*, 9(11):2555–2573, Nov 2014. ISSN 1750-2799. doi: 10.1038/nprot.2014.172. URL https://doi.org/10.1038/nprot.2014.172.

Xiaoyu Li, Zhiguang Zhao, Weina Xu, Rong Fan, Long Xiao, Xuehua Ma, and Zhuo Du. Systems properties and spatiotemporal regulation of cell position variability during embryogenesis. *Cell Reports*, 26(2):313–321.e7, 2019. ISSN 2211-1247. doi: https://doi.org/10.1016/j.celrep.2018.12.052. URL https://www.sciencedirect.com/science/article/pii/S221112471831982X.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *CoRR*, 2017. URL http://arxiv.org/abs/1708.02002v2.

Klas E. G. Magnusson, Joakim Jalden, Penney M. Gilbert, and Helen M. Blau. Global linking of cell tracks using the viterbi algorithm. *IEEE Transactions on Medical Imaging*, 34(4):911–929, 2015. doi: 10.1109/tmi.2014.2370951. URL https://doi.org/10.1109/tmi.2014.2370951.

Caroline Malin-Mayor, Peter Hirsch, Leo Guignard, Katie McDole, Yinan Wan, William C. Lemon, Philipp J. Keller, Stephan Preibisch, and Jan Funke. Automated reconstruction of whole-embryo cell lineages by learning from sparse annotations. *bioRxiv*, 2021. doi: 10.1101/2021.07.28.454016. URL https://www.biorxiv.org/content/early/2021/07/29/2021.07.28.454016.

Pavel Matula, Martin Maška, Dmitry V. Sorokin, Petr Matula, Carlos Ortiz de Solórzano, and Michal Kozubek. Cell tracking accuracy measurement based on comparison of acyclic oriented graphs. *PLOS ONE*, 10(12):e0144959, 2015. doi: 10.1371/journal.pone.0144959. URL https://doi.org/10.1371/journal.pone.0144959.

Katie McDole, Lo Guignard, Fernando Amat, Andrew Berger, Grgoire Malandain, Loc A. Royer, Srinivas C. Turaga, Kristin Branson, and Philipp J. Keller. In toto imaging and reconstruction of post-implantation mouse development at the single-cell level. *Cell*, 175(3):859–876.e33, 2018. ISSN 0092-8674. doi: https://doi.org/10.1016/j.cell.2018.09.031. URL https://www.sciencedirect.com/science/article/pii/S0092867418312431.

Mark W. Moyle, Kristopher M. Barnes, Manik Kuchroo, Alex Gonopolskiy, Leighton H. Duncan, Titas Sengupta, Lin Shao, Min Guo, Anthony Santella, Ryan Christensen, Abhishek Kumar, Yicong Wu, Kevin R. Moon, Guy Wolf, Smita Krishnaswamy, Zhirong Bao, Hari Shroff, William A. Mohler, and Daniel A. Colón-Ramos. Structural and developmental principles of neuropil assembly in c. elegans. *Nature*, 591(7848):99–104, Mar 2021. ISSN 1476-4687. doi: 10.1038/s41586-020-03169-5. URL https://doi.org/10.1038/s41586-020-03169-5.

John Isaac Murray, Zhirong Bao, Thomas J Boyle, Max E Boeck, Barbara L Mericle, Thomas J Nicholas, Zhongying Zhao, Matthew J Sandel, and Robert H Waterston. Automated analysis of embryonic gene expression with cellular resolution in c. elegans. *Nature Methods*, 5(8):703–709, 2008. doi: 10.1038/nmeth.1228. URL https://doi.org/10.1038/nmeth.1228.

Marin Vlastelica Pogani, Anselm Paulus, Vit Musil, Georg Martius, and Michal Rolinek. Differentiation of blackbox combinatorial solvers. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BkevoJSYPB.

N. Ray and S.T. Acton. Active contours for cell tracking. In *Proceedings Fifth IEEE Southwest Symposium on Image Analysis and Interpretation*, pages 274–278, 2002. doi: 10.1109/IAI.2002.999932.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, 2015. URL http://arxiv.org/abs/1505.04597v1.

Josef Lorenz Rumberger, Xiaoyan Yu, Peter Hirsch, Melanie Dohmen, Vanessa Emanuela Guarino, Ashkan Mokarian, Lisa Mais, Jan Funke, and Dagmar Kainmueller. How shift equivariance

impacts metric learning for instance segmentation. *CoRR*, 2021. URL http://arxiv.org/abs/2101.05846v1.

Anthony Santella, Zhuo Du, and Zhirong Bao. A semi-local neighborhood-based framework for probabilistic cell lineage tracing. *BMC Bioinformatics*, 15(1):217, 2014. doi: 10.1186/1471-2105-15-217. URL https://doi.org/10.1186/1471-2105-15-217.

Martin Schiegg, Philipp Hanslovsky, Bernhard X Kausler, Lars Hufnagel, and Fred A Hamprecht. Conservation tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2928–2935, 2013.

Martin Schiegg, Philipp Hanslovsky, Carsten Haubold, Ullrich Koethe, Lars Hufnagel, and Fred A Hamprecht. Graphical model for joint segmentation and tracking of multiple dividing cells. *Bioinformatics*, 31(6):948–956, 2015.

Johannes Schindelin, Ignacio Arganda-Carreras, Erwin Frise, Verena Kaynig, Mark Longair, Tobias Pietzsch, Stephan Preibisch, Curtis Rueden, Stephan Saalfeld, Benjamin Schmid, Jean-Yves Tinevez, Daniel James White, Volker Hartenstein, Kevin Eliceiri, Pavel Tomancak, and Albert Cardona. Fiji: an open-source platform for biological-image analysis. *Nature Methods*, 9(7):676–682, Jul 2012. ISSN 1548-7105. doi: 10.1038/nmeth.2019. URL https://doi.org/10.1038/nmeth.2019.

Nicholas Sofroniew, Talley Lambert, Kira Evans, Juan Nunez-Iglesias, Grzegorz Bokota, Gonzalo Pea-Castellanos, Philip Winston, Kevin Yamauchi, Matthias Bussonnier, Draga Doncila Pop, Ziyang Liu, ACS, Pam, alisterburt, Genevieve Buckley, Andy Sweet, Lorenzo Gaifas, Jaime Rodrguez-Guerra, Lukasz Migas, Volker Hilsenstein, Jordo Bragantini, Gregory R. Lee, Hector, Jeremy Freeman, Peter Boone, Alan R Lowe, Christoph Gohlke, Loic Royer, Andrea PIERR, and Hagai Har-Gil. napari/napari: 0.4.12rc2, October 2021. URL https://doi.org/10.5281/zenodo.5587893.

Ko Sugawara, Cagri Cevrim, and Michalis Averof. Tracking cell lineages in 3d by incremental deep learning. *bioRxiv*, 2021. doi: 10.1101/2021.02.26.432552. URL https://www.biorxiv.org/content/early/2021/02/26/2021.02.26.432552.

Xin Sun, Wei Wang, Dong Li, Bin Zou, and Hongxun Yao. Object contour tracking via adaptive data-driven kernel. *EURASIP Journal on Advances in Signal Processing*, 2020(1):9, Feb 2020. ISSN 1687-6180. doi: 10.1186/s13634-020-0665-x. URL https://doi.org/10.1186/s13634-020-0665-x.

Vladimír Ulman, Martin Maška, Klas E G Magnusson, Olaf Ronneberger, Carsten Haubold, Nathalie Harder, Pavel Matula, Petr Matula, David Svoboda, Miroslav Radojevic, Ihor Smal, Karl Rohr, Joakim Jaldén, Helen M Blau, Oleh Dzyubachyk, Boudewijn Lelieveldt, Pengdong Xiao, Yuexiang Li, Siu-Yeung Cho, Alexandre C Dufour, Jean-Christophe Olivo-Marin, Constantino C Reyes-Aldasoro, Jose A Solis-Lemus, Robert Bensch, Thomas Brox, Johannes Stegmaier, Ralf Mikut, Steffen Wolf, Fred A Hamprecht, Tiago Esteves, Pedro Quelhas, Ömer Demirel, Lars Malmström, Florian Jug, Pavel Tomancak, Erik Meijering, Arrate Muñoz-Barrutia, Michal Kozubek, and Carlos Ortiz de Solorzano. An objective comparison of cell-tracking algorithms. *Nature Methods*, 14(12):1141–1152, 2017. doi: 10.1038/nmeth.4473. URL https://doi.org/10.1038/nmeth.4473.

Martin Weigert, Uwe Schmidt, Tobias Boothe, Andreas Müller, Alexandr Dibrov, Akanksha Jain, Benjamin Wilhelm, Deborah Schmidt, Coleman Broaddus, Siân Culley, Mauricio Rocha-Martins, Fabián Segovia-Miranda, Caren Norden, Ricardo Henriques, Marino Zerial, Michele Solimena, Jochen Rink, Pavel Tomancak, Loic Royer, Florian Jug, and Eugene W. Myers. Content-aware image restoration: pushing the limits of fluorescence microscopy. *Nature Methods*, 15(12):1090–1097, Dec 2018. ISSN 1548-7105. doi: 10.1038/s41592-018-0216-7. URL https://doi.org/10.1038/s41592-018-0216-7.

Zbigniew Wojna, Vittorio Ferrari, Sergio Guadarrama, Nathan Silberman, Liang-Chieh Chen, Alireza Fathi, and Jasper Uijlings. The devil is in the decoder: Classification, regression and gans. *CoRR*, 2017. URL http://arxiv.org/abs/1707.05847v3.

Carsten Wolff, Jean-Yves Tinevez, Tobias Pietzsch, Evangelia Stamataki, Benjamin Harich, Lo Guignard, Stephan Preibisch, Spencer Shorte, Philipp J Keller, Pavel Tomancak, and Anastasios Pavlopoulos. Multi-view light-sheet imaging and tracking with the mamut software reveals the cell lineage of a direct developing arthropod limb. *eLife*, 7:e34410, mar 2018. ISSN 2050-084X. doi: 10.7554/eLife.34410. URL https://doi.org/10.7554/eLife.34410.

Appendix A. Appendix

A.1. Data

Fluo-N3DH-CE The **Fluo-N3DH-CE** dataset (Murray et al., 2008) consists of four 3d+time recordings. It was recorded with a Zeiss LSM 510 Meta microscope with an Plan-Apochromat 63x/1.4 (oil) objective, a voxel size in microns of 0.09 x 0.09 x 1.0 (thus the data is anisotropic) and a frame rate of one frame every 1.5min. For two recordings both the raw image data and the annotations are public, used as the training set. We trained on one of them and validated on the other to determine all hyperparameters and then retrained on both to make use of all training data.

The annotations consist of a few segmented slices that we did not use (each frame consists of a z-stack of 2d xy images; a slice is a single element of such a stack) and the full tracks (point annotations plus connections). The polar bodies are not annotated. For the other two recordings only the raw image data is public (the test set). To get the score on this data the results plus employed software have to be uploaded to the benchmark server. The organizers have to be able to reproduce the results using the software. At the time of submission our method heads the leaderboard for this dataset for the detection score (DET) and tracking score (TRA) (see Table 1).

The challenge evaluates a segmentation score, too. As our model does not produce segmentation results we simulate it by employing the cell indicator prediction. We use the points of detection, as determined by the ILP, as seed points in a seeded watershed. The (inverted) cell indicator map is used as the watershed surface. We threshold the result twofold: Firstly, based on the prediction on the training set we determine a threshold τ for the cell indicator map and use this as a foreground mask. Secondly, we estimate the nuclei size across the time series by roughly measuring their size at up to 5 frames with varying nuclei size and count. We then mask each instance with a sphere (isotropic in world space, anisotropic in object space) centered at its point of detection and with a size equal to the estimated nuclei size based on the next later estimated frame.

No further postprocessing, e.g., to filter very short tracklets, is applied. For the currently to the CTC submitted results we did not filter the polar bodies from the created tracks resulting in some additional false positives. This will be done in the next version.

mskcc_confocal The **mskcc_confocal** dataset consists of three fully annotated 3d+time recordings. It was recorded with a Zeiss LSM 510 confocal microscope, it is anisotropic as well and has a frame rate of one frame/min. The ground truth has been created by using Starrynite (Bao et al., 2006; Santella et al., 2014), followed by manual curation. We use the uncurated results as a baseline. The raw data consists for 425 frames, with the first 370 frames of them being annotated and curated. We train and evaluate all our models on the first 270 frames, in total there are 50k-60k cell detections. At this stage *C. elegans* has around 550 cells, which is close to the maximum number (558 for hermaphrodites). Most divisions have already occurred by the 270th frame and some cells have already undergone apoptosis. In addition we evaluate the same models on the first 200 frames, which is in line with many studies conducted on *C. elegans*. At this stage *C. elegans* has around 350 cells, and all but one round of divisions have occurred (about 20k cell detections)

In future work we will extend it to the full 370 frames. The more the embryonic development progresses the denser the data becomes. Additionally the movement of the worm itself (in this context referred to as twitching) increases significantly.

For each experimental run we use one recording as training data, one for validation and one as our test set. We do this for all six possible combinations of assignment of our recordings to the

three sets, and repeat each one three times with different random seeds, leading to a total of 18 experimental runs. Each number we report is obtained by averaging over the 18 run.

nih_ls The **nih_ls** data was recorded with an ASI diSPIM light sheet microscope (Kumar et al., 2014), the views have been merged and the data deconvolved thus the data is isotropic. The time range of the data is similar to the confocal data, as is the experimental setup. Again we train on the first 270 frames and evaluate on the first 270 frames and the first 200 frames.

A.2. Network Architecture/Training

A.2.1. TRACKING/U-NET

The basic tracking setup follows the approach of Malin-Mayor et al. (2021), with some modifications. As the data is too large to be processed at once, it is processed in tiles. The input size can be adapted depending on the available GPU memory. We follow the rules of Rumberger et al. (2021) to enable seamless predictions. Therefore valid padding is used for all convolutions and the output tiles during inference are cropped correctly before stitching. To exploit temporal context the input of the network are 4d patches, 3d tiles of seven consecutive frames, that are used to compute the output for the center frame. We employ 4d convolutions[2] in the first layers to enable the network to profit from this. For the anisotropic data we do not downsample the depth dimension z in the first pooling layers until the voxel size is approximately isotropic (Heinrich et al., 2018). Separate networks are trained for the cell candidates and the motion vectors in the sense that they have independent weights, however both are trained at the same time on the same batches and information from the cell candidate network is used in the loss computation of the motion vector network.

We downsample three times in total using max pooling, for upsampling we use separable transposed convolutions (Wojna et al., 2017). At each downsample step we increase the number of feature maps fourfold, initially we start with 12. All convolutions have a kernel size of 3 (to be precise 3^d, 3^3 for 3d and 3^4 for 4d convolutions). The network is trained for 400k iterations with a batch size of one. We use stochastic weight averaging (SWA) (Izmailov et al., 2018) every 1k iterations starting after 50k iterations, this significantly boosts performance. We use a large set of base augmentations: elastic deformations, rotation, flipping, resizing, intensity. For the confocal data it proved beneficial to additionally add noise augmentations (salt&pepper and speckle) and a histogram augmentation that varies the strength of the fluorescent nuclei signal by keeping the low intensity pixels stable and varying the height of the intensity bump of high intensity pixels (typically the nuclei). During training we sample random patches: in 10% of cases a completely random location is chosen, in the remaining 90% of cases we chose a location that contains at least some randomly chosen point. The denser the area the less likely a specific point is chosen, otherwise early frames remain under-sampled due to the lower number of cells. As divisions are quite rare we over-sample them, in 25% of these cases one division has to be contained in the patch. One patch typically contains multiple cells. We use the ADAM optimizer with a learning rate of $5\mathrm{e}{-5}$

Detection To detect nuclei we follow the approach of Malin-Mayor et al. (2021) and Höfener et al. (2018). A Gaussian-shaped blob is placed at the location of every ground truth annotation and regressed. The variance is chosen such that the blob approximately covers the respective stained nucleus. If the ground truth annotations contain radii, these are used for this purpose, otherwise a rough estimated radius is used (see Suppl. Sec. A.5). The resulting map is normalized such that

2. https://github.com/funkey/conv4d

the maxima have a value of one. The output of the network is processed with a sigmoid activation function and a simple pixelwise weighted mean squared loss function is used to learn a regression model. We introduce pixelwise weights to highlight the area around nuclei. Pixels with a target value (in the Gaussian map) above a certain threshold have weight 1, all other pixels a very low weight (depending on the dataset, e.g., $1e{-}5$). As the data is 3d, and especially for the isotropic data, the ratio of background pixels to foreground pixels is much higher than for 2d data (cf. surface area of circle and square vs. volume of sphere and cube). With all pixels having the same weight the network can easily degenerate to pure background. Having a weight of zero for the background is a good option for sparse annotations (cf. (Malin-Mayor et al., 2021)), however, this makes it harder to discern false positives as the network has no information for data not in the vicinity of ground truth annotations. As we have dense annotations, in the sense that all nuclei in a frame are annotated, we can make use of this.

During inference we employ a max pooling layer with stride one and a window size slightly smaller than the size of a nucleus to perform non-maximum suppression (NMS) and extract the maxima, our cell candidates.

Motion Vector Additionally we learn to predict the motion of a cell between adjacent time frames. Following Malin-Mayor et al. (2021), and similarly to Hayashida et al. (2020), we predict a vector pointing backwards in time to the position in the previous time frame. For our 3d data the output of this network is thus a three dimensional vector. We apply no activation function as the vector is, in principle, unrestricted. Again, a weighted mean squared loss function is used. However the weights are computed differently. At the start of the training each pixel inside a nucleus has weight one, and outside weight zero. At the end of the training only the pixel with the maximum value in the cell indicator map has weight one, the others zero. As the latter is very sparse we, to ease the learning difficulty, we start with the former and blend these two together, the interpolation factor increases smoothly from zero to one. The in the annotations included or estimated (see Sec. A.5) nucleus radius is used to define inside vs. outside.

During inference we extract the motion vectors corresponding to our cell candidates from detection network.

A.2.2. CELL STATE/RESNET

Similarly to Santella et al. (2014) we use a classifier to determine the cell state of each detection. We assign to each detection one of four classes: parent cell (cell that is about to undergo cell division), daughter cell (cell that just divided), polar body and none of those (continuation). We use a ResNet-18 with 3d convolutions and bottleneck blocks for this task. The input is a mini-batch of patches. Each patch is smaller than the U-Net input (8x64x64 (anisotropic) or 64x64x64 (isotropic) pixels) and is centered on a single cell, though neighboring cells around it can be contained partly. To capture temporal context each patch contains 5 3d patches, from the 2 time frames before and after the cell in question. For this network we do not employ 4d convolutions but interpret them as multiple input channels. We use global average pooling at the end followed by a 1x convolution to the appropriate number of output neurons to get the output. For the **Fluo-N3DH-CE** and the **nih_ls** datasets we use a standard cross entropy loss, for the **mskcc_confocal** dataset the focal loss (Lin et al., 2017).

As the vast majority of detections belong to the continuation class we oversample instances of the other three classes in each mini batch to reduce bias towards the majority class. Yet having

each batch consist of equal parts of each class led to increased false positives. We determined empirically on the validation set that $\frac{3}{6}$ from the continuation class and $\frac{1}{6}$ each from the other classes balances false positives and false negatives for the minority classes nicely. The scores for the parent/daughter/continuation classes are incorporated as costs into the ILP (see Sec. 3.2 and Suppl. Sec. A.3). The score for the polar body class is used to optionally remove polar body tracks (see Suppl. Sec. A.8). For a visual example of polar bodies see Suppl. Figure 2.

We train the network for up to 80k iterations performing early-stopping. We perform the same augmentations as for the U-Net. We use the ADAM optimizer with a learning rate of $5\mathrm{e}{-4}$, lowered to $5\mathrm{e}{-6}$ after 20k iterations. For use in the ILP we execute the trained network on each candidate detection.

A.3. ILP-based Linking

Following Malin-Mayor et al. (2021), to represent a solution candidate y we first create main indicator variables for all nodes y_{nodes} and edges y_{edges}. A valid solution is fully represented by these, though to facilitate our constraints we add additional dependent auxiliary indicators: A track cost indicator y_{track} per node to mark nodes that start a track (Tracks starting in the first frame are free, and in a perfect solution no other tracks should start as all cells are connected by divisions). Parent/daughter/continuation indicators $y_{parent}, y_{daughter}, y_{continuation}$ per node to mark the cell state (in line with the cell state classifier). We define a target indicator vector:

$$y = \left[y_{nodes}, y_{track}, y_{parent}, y_{daughter}, y_{continuation}, y_{edges} \right]^T \in \{0,1\}^{5|V|+|E|}$$

To compute our costs we construct a sparse feature matrix $S^{dim(y)\times dim(w)}$ with one row per indicator. The columns are: A node selection constant that is 1 for *node* indicators and 0 otherwise. A node score that is equal to the cell indicator score s_v for *node* indicators and 0 otherwise. An edge score that is, given some edge $e = (v, u)$, equal to the distance between the predicted position $\hat{p}_v$ and the actual position p_u of the node u for *edge* indicators and 0 otherwise. A track cost score that is 1 for *track_start* indicators and 0 otherwise. A division constant that is 1 for *parent class* indicators and 0 otherwise. A parent score that is equal to the *parent class* prediction for *parent* indicators and 0 otherwise. A daughter score that is equal to the *daughter class* prediction for *daughter* indicators and 0 otherwise. A continuation score that is equal to the *continuation class* prediction for *continuation* indicators and 0 otherwise.

Finally, we create a weight vector with one tunable hyperparameter per column:

$$w = \left[w_{node\_selection}, w_{node\_score}, w_{edge\_score}, w_{track\_cost}, w_{division}, w_{parent}, w_{daughter}, w_{continuation} \right]^T$$

Following Malin-Mayor et al. (2021) we use Gurobi (Gurobi Optimization, 2021) for block-wise solving. For computational reasons the ILP is solved in blocks. We divide the whole recording into non-overlapping blocks. The method itself is agnostic to the axes of division, the blocks can be divided both spatially and temporally. This is necessary for large organisms, for *C. elegans* though it is sufficient to divide it along the temporal axis. To ensure consistency each block takes some amount of context around it into account, leading to overlapping blocks. If a neighboring block has already been computed, its results are adhered to and incorporated via constraints into the local

block ILP. Blocks can be computed in parallel as long as they are not adjacent. However, this block-based processing means that the final tracks will not necessarily be the globally optimal tracks, they might just approximate them. We argue that due to the temporal nature of the data, events in temporally distant frames can be resolved independently without a significant loss in accuracy.

A.4. Optimization Constraints

We add a number of different constraints to the ILP to ensure the validity and consistency of the resulting tracks. The base constraints according to Malin-Mayor et al. (2021) are:

For each edge $e = (u, v)$ the edge consistency constraint $2y_{edge_e} - y_{node_v} - y_{node_u} \leq 0$ ensures that if an edge is selected its endpoints have to be selected as well.

To encourage temporal continuity one constraint per node is added: Let P_v be the set of edges from node v at t_v to nodes in $t_v - 1$. The constraint $\sum_{p \in P_v}(y_{node_p}) + y_{track_v} - y_{node_v} = 0$ ensures that if node v is selected either its track indicator is set (signaling that this node is the start of a track which comes with a cost attached) or there is exactly one edge to a node in the previous time frame.

To ensure biologically moral validity we add a constraint on cell divisions per node: Let N_v be the set from nodes in $t_v + 1$ to a node v at t_v. The constraint $\sum_{n \in N_v}(y_{node_n}) - 2y_{node_v} \leq 0$ ensures that for each selected node there can be at most two edges to the next time frame.

Simple equality constraints are added to ensure block-wise consistency. If a neighboring block has already been computed its decisions for edges e in the overlap area are accepted: $y_{edge_e} = 0$ or $y_{edge_e} = 1$ respectively.

In addition we add constraints to incorporate the learnt cell state classifier scores into the ILP. The goal is to encourage the optimizer to select indicators in line with the state scores by lowering the cost in case of agreement and increasing the cost in case of disagreement. The cell state constraints are: For each edge $e = (u, v)$ the constraints $y_{daughter_u} + y_{edge_e} - y_{parent_v} \leq 1$ and $y_{parent_v} + y_{edge_e} - y_{daughter_v} \leq 1$ ensure that if the endpoint v at time t_v of a selected edge has y_{parent_v} selected than the endpoint u at time $t_v + 1$ has to have $y_{daughter_u}$ selected and vice versa. In addition each selected node v has have either y_{parent_v}, $y_{daughter_v}$ or $y_{continuation_v}$ active, enforced via $y_{parent_v} + y_{daughter_v} + y_{parent_v} - y_{node_v} = 0$.

A.5. Radius Estimation

If no radius information is included in the annotations (such as in the **Fluo-N3DH-CE** data) we estimate it roughly with a handful manual measurements. We open a few frames (across the whole time line) in an appropriate image viewer (e.g., ImageJ/Fiji (Schindelin et al., 2012)) and measure the current nucleus radius at a few representative cells. We then assign this radius to all cells in all frames before this one (but after the previously examined frame). The radius is used both during training to determine the variance for the regressed Gaussian blobs and the inside vs. outside of cells for the weights in the motion vector loss and for the necessary segmentation in the computation of the DET and TRA scores.

A.6. Foreground Mask

To remove spurious detections completely outside of an embryo we create a simple foreground mask. We open a frame somewhere in the middle of the timeline in an appropriate viewer (e.g., napari (Sofroniew et al., 2021)) and draw a rough 2d polygon around it that is extended in the depth

dimension z. All detections outside this area are discarded. If many samples have to be processed the mask creation can also be automated by a combination of computer vision/morphology operations (blurring, thresholding, hole-filling), however this failed in the case of the **Fluo-N3DH-CE** data due to some artifacts contained in the data.

A.7. Polar and Apoptotic Bodies

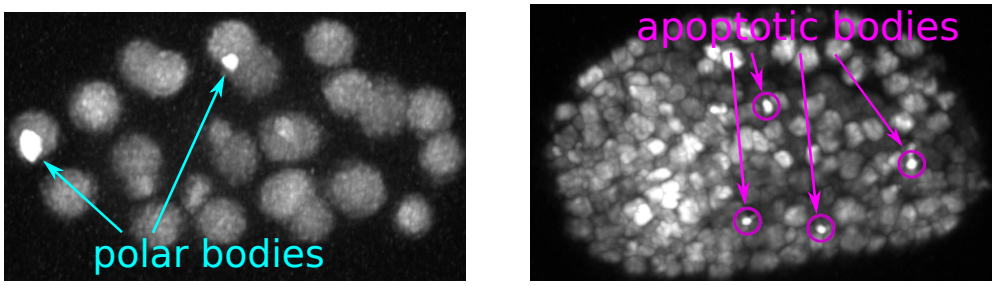

Figure 2: Examples of polar and apoptotic bodies in *C. elegans*

A.8. Postprocessing

Depending on the type of study the polar bodies are either of interest or should be removed. We manually created polar bodies annotations for the training of our cell state classifier using MaMuT (Wolff et al., 2018). If they are to be removed, we divide our tracks into chains by temporarily removing all connections from daughter cells to their parent cell. We then perform a majority voting per chain and remove all chains where the majority of cells were classified as a polar body. However, they are not removed in the current CTC results, this will happen in a future submission. No further postprocessing is performed.

A.9. Experiments

Table 4: Quantitative results for **mskcc_confocal** and **nih_ls** on both the first 200 frames (approx. 350 cells in last frame) and the first 270 frames (approx. 550 cells in last frame), *FP*: false positive edge, *FN*: false negative edge, *IS*: identity switch/cross-over of tracks, *FP-div*: false positive division, *FN-div*: false negative division, *total div*: sum of division errors, *total*: sum of all errors, normalized per 1k edges. *REFT*: ratio error free tracks (number of cells in the last frame whose reverse track has no error divided by the number of GT cells in the last frame, biased in favor of fp as superfluous tracklets ending earlier are not counted). *ours wo/cls* matches the prior work (Malin-Mayor et al., 2021).

	FP	FN	IS	FP div	FN div	total div	total	REFT
mskcc_confocal 200 frames								
Starrynite	7.703	5.470	0.220	0.375	1.240	1.614	15.006	0.878
ours wo/cls	5.259	2.477	0.024	0.343	0.139	0.484	8.244	0.956
ours ssvm	5.719	1.647	0.026	0.036	0.595	0.631	8.031	0.934
mskcc_confocal 270 frames								
Starrynite	7.902	13.321	0.618	0.579	1.184	1.763	22.324	0.769
ours wo/cls	4.575	5.464	0.055	1.122	0.261	1.382	11.659	0.891
ours ssvm	3.684	5.703	0.048	0.066	0.513	0.579	9.909	0.839
nih_ls 200 frames								
ours w/cls	2.522	2.754	0.076	0.291	0.426	0.716	6.069	0.941
nih_ls 270 frames								
ours w/cls	12.047	7.193	0.446	0.317	0.351	0.668	20.353	0.852

Table 5: Ablation study, on both the first 200 frames (approx. 350 cells in last frame) and the first 270 frames (approx. 550 cells in last frame) of the **mskcc_confocal** data. We ablate solving an ILP altogether (ILP), incorporating the cell state classifier (cls), employing an sSVM for hyperparameter search (ssvm), and incorporating the polar body filter (pbf). Description of error types see Table 2.

ILP	cls	ssvm	pbf	FP	FN	IS	FP div	FN div	total div	total	REFT
mskcc_confocal 200 frames											
✗	✗	✗	✗	7.032	1.032	0.016	0.644	0.139	0.783	8.863	0.967
✓	✗	✗	✗	5.259	2.477	0.024	0.343	0.139	0.484	8.244	0.956
✓	✓	✗	✗	5.037	1.876	0.016	0.094	0.242	0.336	7.265	0.954
✓	✓	✓	✗	5.719	1.647	0.026	0.036	0.595	0.631	8.031	0.934
✓	✓	✓	✓	3.016	1.714	0.029	0.032	0.565	0.594	5.356	0.932
mskcc_confocal 270 frames											
✗	✗	✗	✗	5.008	4.639	0.048	1.600	0.255	1.855	11.550	0.912
✓	✗	✗	✗	4.757	5.464	0.055	1.122	0.261	1.382	11.659	0.891
✓	✓	✗	✗	3.408	5.669	0.028	0.202	0.263	0.464	9.570	0.871
✓	✓	✓	✗	3.684	5.703	0.048	0.066	0.513	0.579	9.909	0.839
✓	✓	✓	✓	2.556	5.717	0.050	0.062	0.510	0.573	8.907	0.836

Table 6: For comparison with other methods we also report node and edge recall. However their informative value is limited as the values are quite saturated, still, there is some improvement.

	node recall	edge recall
mskcc_confocal 200 frames		
Starrynite	0.9961	0.9827
ours wo/cls	0.9981	0.9833
ours ssvm	0.9993	0.9850
mskcc_confocal 270 frames		
Starrynite	0.9927	0.9851
ours wo/cls	0.9988	0.9920
ours ssvm	0.9991	0.9923
nih_ls 200 frames		
ours w/cls	0.998	0.988
nih_ls 270 frames		
ours w/cls	0.998	0.994

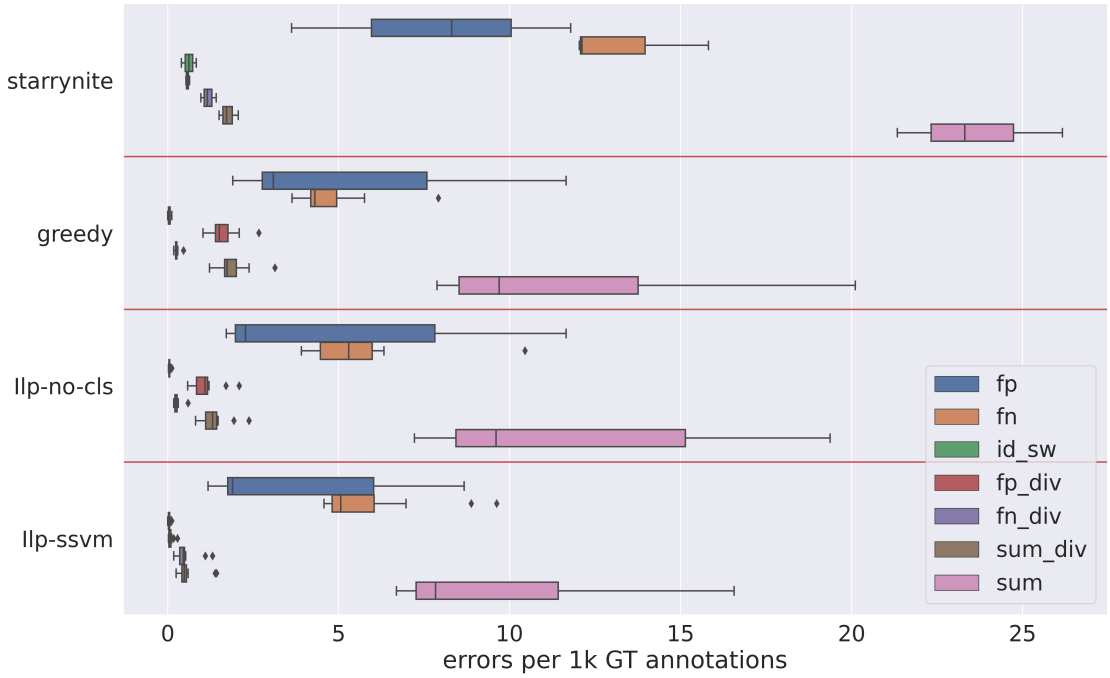

Figure 3: Box and whisker plot of the errors of the different approaches on the **mskcc_confocal** data. Starrynite (Bao et al., 2006) is an often used method in the analysis of *C. elegans* tracks. *greedy* refers to our method without the ILP (and thus without the cell state classifier). *Ilp-no-cls* is our method without the cell state classifier and with grid-searched hyperparameters (matches the prior work (Malin-Mayor et al., 2021)). *Ilp-ssvm* is the full method with automatically determined hyperparameters. *fp* are false positive edges, *fn* are false negative edges, *id_sw* are identity switches (cross-over of tracks), *fp_div* are false positive (superfluous) divisions, *fn_div* are false negative (missing) divisions, *sum_div* is the sum of wrong divisions and *sum* the sum of all errors. All numbers are normalized per 1000 ground truth annotations. Divisions that are off by one frame compared to the annotations are not counted as errors as the limited frame rate leads to inherent inaccuracies in the data and annotations. Each step lowers the number of errors. *greedy* lowers especially the number of *fp* and *fn* edges, not as much the number of false divisions. The ILP on its own (*Ilp-no-cls*) can already lower the number of false divisions a bit, but the inclusion of the classifier in *Ilp-ssvm* lowers them drastically. For the numbers see Table 4 and Table 5.

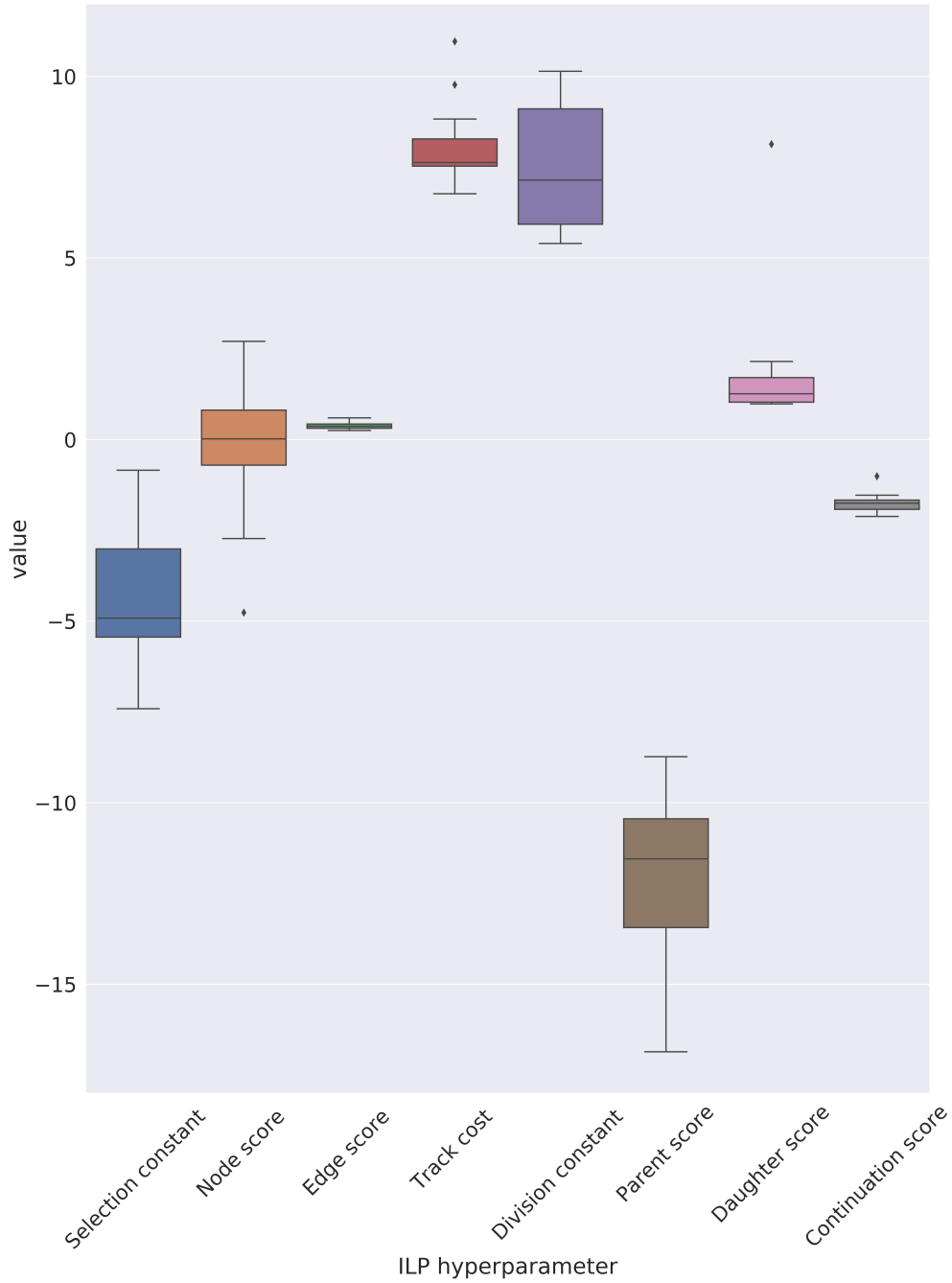

Figure 4: Box and whisker plot of the distribution of the automatically determined ILP hyperparameters over the 18 experimental runs of the **mskcc_confocal** dataset. The sSVM finds similar values for each respective candidate graph and with a similar ratio to each other.