Network Flow Integer Programming to Track Elliptical Cells in Time-Lapse Sequences

Engin Türetken*, Xinchao Wang*, Carlos J. Becker, Carsten Haubold, and Pascal Fua, *Fellow, IEEE*

Abstract—We propose a novel approach to automatically tracking elliptical cell populations in time-lapse image sequences. Given an initial segmentation, we account for partial occlusions and overlaps by generating an over-complete set of competing detection hypotheses. To this end, we fit ellipses to portions of the initial regions and build a hierarchy of ellipses, which are then treated as cell candidates. We then select temporally consistent ones by solving to optimality an integer program with only one type of flow variables. This eliminates the need for heuristics to handle missed detections due to partial occlusions and complex morphology. We demonstrate the effectiveness of our approach on a range of challenging sequences consisting of clumped cells and show that it outperforms state-of-the-art techniques.

Index Terms—Cell tracking, integer programming, network flows

1 Introduction

Detecting and tracking cells over time is key to understanding cellular processes including division (mitosis), migration, and death (apoptosis). Modern microscopes produce vast image streams making manual tracking tedious and impractical. High-throughput automated systems are therefore increasingly in demand and several cell tracking competitions have recently been organized to attract Computer Vision researchers' interest and speed up progress [4], [5]. These competitions have shown that state-of-the-art methods are still error-prone due to occlusions, imaging noise, and complex cell morphology.

Cell tracking is an instance of the more generic multi-target tracking problem with the additional difficulties that cells, unlike for example pedestrians, can either divide or wither away and disappear in mid-sequence. In its generic form, the problem is often formulated as a two-step process that involves first detecting potential objects in individual frames and then linking these detections into complete trajectories. This approach is attractive because spurious detections, such as false positives due to imaging noise and artifacts, can be eliminated by imposing temporal consistency across many frames, which is more difficult to do in recursive approaches.

Many of the most successful algorithms to people [6], [7], [8], [9], [10], [11] , cell tracking [12], [13], [1] and 3D reconstruction [14], [15] follow this two-step approach by first running an object detector on each frame independently and building a graph whose nodes are the detections and edges connect pairs of them. They then find a subgraph that represents object trajectories by considering the whole graph at once. However, to handle missed detections, these methods often rely on heuristic procedures that offer no guarantee of optimality. This is of particular concern for cell tracking because detections are often unreliable due to the complex morphology of cell populations. For example, in Fig. 1, groups of cells that appear clumped together in some frames can only be told apart when considering the sequence as a whole.

The approach proposed in [2] addresses this issue by introducing a factor graph for joint cell segmentation and tracking. Inference is then achieved by solving a relatively complex integer program that involves several types of variables and constraints. The algorithm starts from a watershed-based oversegmentation that is sensitive to inaccuracies in pixel probability estimates. It is therefore subject to errors when the cells are clumped together and hard to distinguish from each other.

In this paper, we formulate joint detection and tracking in terms of constrained maximum a posteriori estimation. This lets us cast the problem as an integer program expressed in terms of a single type of flow variables. We focus on tracking elliptical cells in time-lapse image sequences. To explicitly account for the often ambiguous output of cell detectors, we devised a robust ellipse-fitting algorithm to generate multiple and potentially conflicting initial hypotheses from the output of a foreground-background segmentation algorithm, as depicted by Fig. 2. We then model critical cell events, such as migration and division, using flow variables and resolve the conflicts by solving the resulting integer program, which is conceptually much simpler than those of earlier approaches [13], [1], [16], [2] and yields fewer variables and constraints. We will refer to our approach as tracking Elliptical Cells using network fLow Integer Programming (**ECLIP**).

Although network flow formulation has been used in this context before, existing approaches [17], [18] focus on cell tracking in consecutive image pairs, while our approach aggregates image evidences from the whole sequence and conducts global optimization over all potential cell locations at all time frames. Furthermore, we handle competing hypotheses within the same optimization framework instead of having to decide at detection time, as in [16], [2].

* *The authors contributed equally.*

- *E. Türetken is with the Swiss Center for Electronic and Microtechnology, Neuchâtel CH-2002, Switzerland. E-mail:engin.tueretken@alumni.epfl.ch*
- *X. Wang is with the Image Formation and Processing Group (IFP), Beckman Institute, University of Illinois Urbana-Champaign (UIUC), Urbana Illinois 61801. E-mail: xinchao@illinois.edu*
- *C. Becker and P. Fua are with the Computer Vision Laboratory, School of Computer and Communication Science, École Polytechnique Fédérale de Lausanne (EPFL), Lausanne CH-1015, Switzerland. E-mail: {carlos.becker, pascal.fua}@epfl.ch*
- *C. Haubold is with the Heidelberg Collaboratory for Image Processing, University of Heidelberg, Germany. E-mail: carsten.haubold@iwr.uni-heidelberg.de*

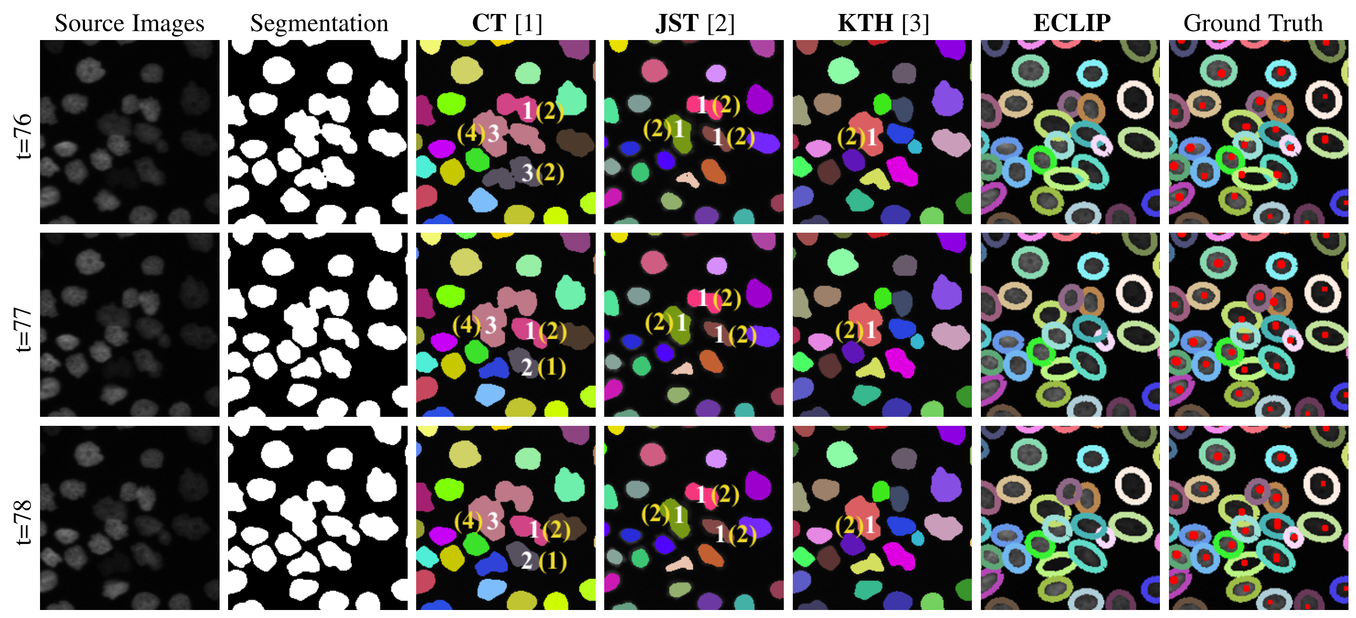


Fig. 1: Three images from a typical sequence; the original segmentations produced by a pixel-based classifier; the results of [1](**CT**), [2](**JST**), [3](**KTH**), and our method (**ECLIP**); the manually annotated tracking ground truth in red dots and the optimal ellipse-tracks obtained using this ground truth. The track identities are encoded in colors. Our approach correctly tracks the cells in spite of long-term segmentation failures, and produce results that are very similar to the ground truth. In the **CT**, **JST** and **KTH** columns, the white-colored numbers indicate the tracker-inferred numbers of cells that are contained by the segments beneath them, and the yellow-colored numbers show the ground truth. Best viewed in color.

Our contribution is therefore a novel approach to simultaneously detecting and tracking cells by first generating an overcomplete set of conflicting cell candidates and then selecting temporally consistent ones. To this end, we introduce a simple yet effective network flow integer programming formulation. We show that **ECLIP** improves trajectories and yields superior detection performance on various datasets as compared to recent approaches [1], [19], [20], [2], including the ones that performed best on the above-mentioned cell-tracking challenges [4], [5].

2 Related Work

Current tracking approaches can be divided into *Tracking by Model Evolution* and *Tracking by Detection* [4]. We briefly discuss state-of-the-art representatives of these two classes below and refer the interested reader to much more complete surveys [21], [4], [22] for further details.

2.1 Tracking by Model Evolution

Most algorithms in this class simultaneously track and detect objects greedily from frame to frame. This means extrapolating results obtained in earlier frames to process the current one, which can be done at a low computational cost and is therefore fast in practice. Such methods have attracted attention both in the cell tracking field [23], [24], [25], [26] as well as in the more general object tracking one [27], [28], [29], [30], [31]. Common techniques in the cell tracking category involve evolving appearance or geometry models from one frame to the next, typically done using active contours [23], [24], [25], [26] or Gaussian Mixture Models [19]. Even though these methods are attractive and mathematically sound, performance suffers from the fact that they only consider a restricted temporal context and therefore cannot guarantee consistency over a whole sequence.

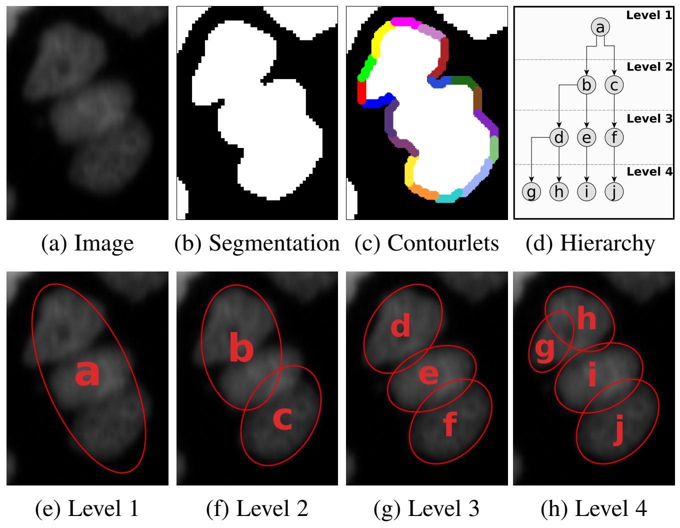


Fig. 2: Hierarchy of detection hypotheses. (a) An image region containing three HeLa cells clumped together. (b) Applying a pixel classifier results in under-segmentation, in which the three cells appear as a single connected component. (c) Automatically extracted contourlets for this component. Each one is overlaid in a different color. They are used to fit ellipses using a hierarchical agglomerative clustering algorithm. (d) The resulting hierarchy of 10 hypotheses. We show only the first four levels for simplicity. (e-h) Individual levels with one, two, three and four hypotheses.

Active contour methods such as [32], [33] address this limitation by considering image batches to segment the cells and recover trajectory fragments. This increases robustness but does not achieve global optimality over long sequence.

2.2 Tracking by Detection

Approaches in this class have proved successful at both tracking people [6], [34], [7], [9], [10], [35], [36], [37], tracking cells [12], [13], [1], and tracking more generic particles including molecules and viruses [38].

They involve first detecting the target objects in individual frames and then linking these detections to produce full trajectories. This involves simultaneously processing more frames than is typically done by Tracking by Model Evolution approaches. Usually, it is also more robust because trajectories are computed by minimizing an objective function that enforces consistency of appearance, disappearance, and division over a batch of frames. This can be seen in the benchmark of [4] in which these methods tend to dominate.

One way to perform tracking-by-detection is to reason in the full spatio-temporal grid formed by stacking up all possible spatial locations over time [39], [40], [41], [42], [43], [44]. This can be done optimally in polynomial time for non-dividing objects such as pedestrians and has made this approach competitive despite the large size of the graphs involved. For dividing objects, the corresponding optimization problem becomes NP-Hard and, to the best of our knowledge, this approach has not been pursued in this context.

Instead, practical tracking-by-detection algorithms for cells rely on a small number of strong detections that can be later linked into complete trajectories. Recent efforts have focused on solving two main challenges specific to cell-tracking, which we discuss below.

Division and disappearance. Cells can divide or die and disappear. While rule-based approaches have been used to handle this, most recent ones formulate tracking as a global IP [13], [1]. This makes it possible to use priors for cell migration, division and disappearance between adjacent frames. One notable exception is the method of [3] that relies on the Viterbi algorithm to sequentially add trajectories to a cell lineage tree. Motion, division, and disappearance are encoded through a scoring function that quantifies how well the lineage tree explains the data. This algorithm scored highest in one of the benchmarks [4] and we will use it as one of our baselines in Section 4. The approach of [45], on the other hand, decouples tracking division events and migration events by excluding the dividing cells from the whole graph. This approach is computationally efficient but makes global optimality elusive by decoupling events that are closely related. The approach of [46], however, focuses on cell segmentation and tracking in the confined nanowell environment. Despite its high throughput, it assumes that the cells do not divide and is therefore not suitable for general-purpose cell tracking.

Clumped cells. There is no guarantee that individual cells will be detected as separate entities in any given frame because two or more cells can clump together, producing under-segmentation errors. Many heuristics have been proposed to solve this problem. One is to assume that clumped cells are unlikely to happen for a specific modality or that they do not pose a problem for the tracker [12], [13], [3]. While this is appropriate in some cases, it is clearly invalid for certain modalities such as the one shown in Fig. 1. Other heuristics involve splitting segmentations using the Radon and watershed transforms [23], [4]. Unfortunately, they are still relatively prone to over- and under-segmentation that complicate the tracking task. The approach of [47] generates oversegmentation by fitting ellipses on Hessian Images. However, this approach focuses on developing embryo and relies on strong assumptions, such as assuming that the number of cells do not decrease from time t to $t+1$.

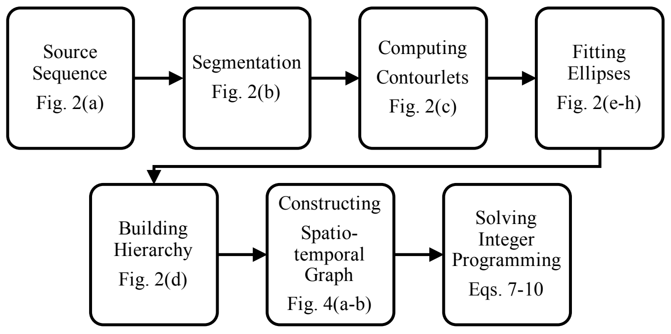

Source Sequence Fig. 2(a)
Segmentation Fig. 2(b)
Computing Contourlets Fig. 2(c)
Fitting Ellipses Fig. 2(e-h)
Building Hierarchy Fig. 2(d)
Constructing Spatio-temporal Graph Fig. 4(a-b)
Solving Integer Programming Eqs. 7-10

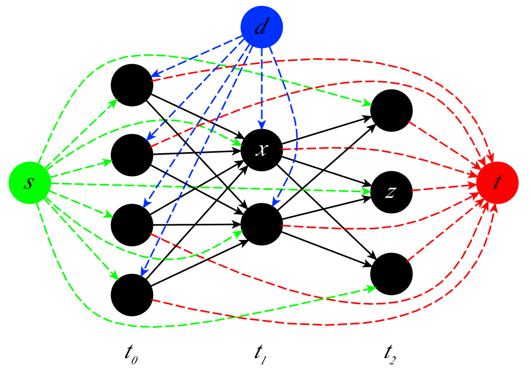
Fig. 3: Workflow of our approach.

An ingenious approach is that of [1], which first finds trajectories by treating segmentations in each frame as clumps of one or more cells, with the exact number being initially unknown, and then uses a factor graph to resolve this ambiguity. However, since the final linking is decoupled from the cell cluster tracking, the cell count inaccuracies may propagate to the final trajectories. The approach of [48] relies on an iterative post-processing step to handle under- and over-segmentation, as in [1]. Therefore, there is no guarantee that a global optimum will be found. To overcome this problem, the algorithm of [2] generates multiple hypotheses by over-segmenting the image into superpixels and subsequently merging them to create competing explanations. Similarly, the algorithm of [49] uses component trees and graph cut to generate the hypotheses. However, neither method proposes spatially overlapping hypotheses, which may result in irrecoverable segmentation errors. By contrast, our approach naturally handles such cases, as depicted by Fig. 2. Furthermore, as we will discuss in Section 3.2, our formulation provides a more compact integer program with much less constraints and variables.

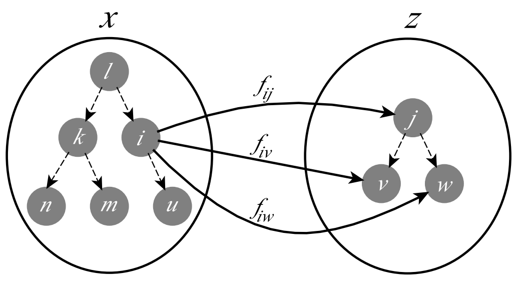
3 METHOD

Fig. 3 depicts our method's workflow, which includes the following steps. Given a source image such as the one of Fig. 2(a), we first use a classifier trained on a few hand-annotated segmentations to produce a binary image of the underlying cell populations. We then look for the best possible description of each connected component in terms of a variable numbers of ellipses. This yields a hierarchy of conflicting interpretations for each component, as depicted by Fig. 2(e–h), which we organize into a directed spatio-temporal graph, such as the one of Fig. 4. Its nodes are individual ellipses, its edges connect nearby ones in consecutive frames, and it features exclusion sets that denote incompatible hypotheses. Full trajectories are obtained by solving an integer program with a small number of constraints to exclude incompatible hypotheses and enforce consistency while allowing for cell-division, migration and death.

In this section, we first describe our approach to segmenting cell images and building hypotheses graphs from them. We then formulate the simultaneous detection and tracking problem on these graphs as a constrained network flow problem, and discuss how we compute the various energy terms of its objective function.



(a)



(b)

Fig. 4: Spatio-temporal graph of hypotheses for 3 consecutive time frames. (a) Each black circle denotes a hypervertex corresponding to a connected component of the segmentation and is connected to neighboring ones at the next time step. The special vertices s (source, in green), t (sink, in red) and d (division, in blue) allow for cell appearance, disappearance and division, respectively. (b) Each hypervertex, such as x and z, contains a hierarchical set of hypotheses (vertices) such as those depicted by Fig. 2. These hypotheses are shown as gray circles and connected to nearby ones in the following frame via directed edges. To avoid clutter, we only show three of these edges. For each hypervertex, we define an exclusion set for each leaf vertex. For example, we define three exclusion sets on x: $S_n = \{l, k, n\}$, $S_m = \{l, k, m\}$ and $S_u = \{l, i, u\}$. We allow the tracker to select at most one vertex within each one. Best viewed in color.

3.1 Building Hierarchy Graphs

Our algorithm, like those of [12], [13], [1], starts by segmenting cells using local image features. To this end, we first train the binary random forest pixel classifier of [50] for each evaluation dataset on a few partially annotated images. We use four different kinds of low-level features: pixel intensities, gradient and Hessian values, and difference of Gaussians, all of which are computed by first smoothing the input image using Gaussians of standard deviations $\{2, 4, 8, 16, 32, 64\}$.

Applying the resulting classifier to the full image sequences results in segmentations which often contain groups of clumped cells, such as the ones shown in Fig. 1. Therefore, each connected component of the segmentation potentially contains an a priori unknown number of cells.

We produce a hierarchy of conflicting detection hypotheses for each such component by fitting a variable number of ellipses to its contours. More specifically, we first identify all the contour points that are local maxima of curvature magnitude. This is done iteratively by selecting the maximum curvature points and suppressing their local neighborhoods. We then break the contour into *contourlets*, that is, short segments between these points, as shown in Fig. 2(c). We cluster them in a hierarchical agglomerative manner and fit ellipses to each resulting cluster using a non-iterative least squares approach of [51]. In all our experiments, we set the size of the suppression neighborhood to seven pixels because this is the minimum number of points required to reliably fit an ellipse using this approach.

Agglomerative clustering requires defining a distance between the objects to be clustered, in this case subsets of the contourlets derived from a given connected component. Let C be the set of contourlets and C_i and C_j two of its subsets. We take their distance to be

$$\left[\sum_{c_l \in \{C_i \cup C_j\}} h(c_l, e) + \sum_{c_l \in C \setminus \{C_i \cup C_j\}} g(c_l, e)\right] c(e) \sqrt{\frac{1}{1 + (ecc(e))^2}}\,, \quad (1)$$

where e is the ellipse obtained by fitting to the contourlets of $C_i \cup C_j$, and $c(e)$ and $ecc(e)$ are its circumference and eccentricity respectively. $h(c_l, e)$ denotes the Hausdorff distance between contourlet c_l and ellipse e, while $g(c_l, e)$ is the Hausdorff distance computed using only points in c_l that fall within e. In other words, let $\hat{c}_l$ be the set of points on c_l that fall within e. We have $g(c_l, e) = h(\hat{c}_l, e)$. The first term in this 3-term product captures image evidence along the contours while the other two are shape regularizers designed to prevent implausible ellipse geometries from appearing in the solution.

Given the resulting ellipse hierarchies in successive images, we build a graph whose vertices are the ellipses and edges link pairs of them that belong to two spatially close connected components in consecutive frames. In our experiments, we set a distance threshold of 25 pixels, and link pairs of ellipses within this distance. The resulting graph has a hierarchical structure that allows for finding the globally optimal cell detections and trajectories in a single shot, as discussed below.

3.2 Network Flow Formalism

The procedure described above yields a directed graph $G' = (V', E')$, which we then augment with three distinguished vertices, a source s, a sink t, and a division node d, as depicted by Fig. 4. We connect these three vertices to every other vertex in G' to allow for cell appearance, disappearance, and division in mid-sequence. Note that, we consider both the cell birth and migration into the image as a cell appearance event. Similarly, we treat cell death and migration out of the image as a disappearance event.

Let $G = (V, E)$ be the resulting graph obtained after the augmentation. We define a binary flow variable f_{ij} for each edge $e_{ij} \in E$ to indicate the presence of any one of the following cellular events:

- Cell migration from vertex i to vertex j,
- Appearance at vertex j, if $i = s$,
- Division at vertex j, if $i = d$,
- Disappearance at vertex i, if $j = t$.

Let $\mathbf{f}$ be the set of all f_{ij} flow variables and $\mathbf{F} = \{F_{ij}\}$ be the set of all corresponding hidden variables. Given an image sequence $\mathbf{I} = (\mathbf{I}^1, \ldots, \mathbf{I}^T)$ with T temporal frames and the corresponding graph G, we look for the optimal trajectories $\mathbf{f}^*$ in G as the solution of

$$\begin{aligned}
\mathbf{f}^* &= \operatorname*{argmax}_{\mathbf{f}\in\mathcal{F}} P(\mathbf{F}=\mathbf{f} \mid \mathbf{I}) && (2)\\
&\approx \operatorname*{argmax}_{\mathbf{f}\in\mathcal{F}} \prod_{e_{ij}\in E} P(F_{ij}=f_{ij} \mid \mathbf{I}^{t(i)},\mathbf{I}^{t(j)}) && (3)\\
&= \operatorname*{argmax}_{\mathbf{f}\in\mathcal{F}} \prod_{e_{ij}\in E} P(F_{ij}=1 \mid \mathbf{I}^{t(i)},\mathbf{I}^{t(j)})^{f_{ij}} \times \\
&\qquad\qquad P(F_{ij}=0 \mid \mathbf{I}^{t(i)},\mathbf{I}^{t(j)})^{(1-f_{ij})} && (4)\\
&= \operatorname*{argmax}_{\mathbf{f}\in\mathcal{F}} \sum_{e_{ij}\in E} \log\left(\frac{P(F_{ij}=1 \mid \mathbf{I}^{t(i)},\mathbf{I}^{t(j)})}{P(F_{ij}=0 \mid \mathbf{I}^{t(i)},\mathbf{I}^{t(j)})}\right) f_{ij} && (5)\\
&= \operatorname*{argmax}_{\mathbf{f}\in\mathcal{F}} \sum_{e_{ij}\in E'} \log\left(\frac{\rho_{ij}}{1-\rho_{ij}}\right) f_{ij} + \sum_{j\in V} \log\left(\frac{\rho_a}{1-\rho_a}\right) f_{sj}\\
&\quad + \sum_{i\in V} \log\left(\frac{\rho_d}{1-\rho_d}\right) f_{it} + \sum_{j\in V} \log\left(\frac{\rho_j}{1-\rho_j}\right) f_{dj}\ , && (6)
\end{aligned}$$

where $\mathbf{I}^{t(i)}$ is the temporal frame containing vertex i, and $\mathcal{F}$ denotes the set of all feasible cell trajectories, which satisfy linear constraints. In Eq. 3, we assume that the flow variables F_{ij} are conditionally independent given the evidence from consecutive frame pairs. Eqs. 4 and 5 are obtained by using the fact that the flow variables are binary and by taking the logarithm of the product. Finally, in Eq. 6, we split the sum into four parts corresponding to the four events mentioned above.

The appearance and disappearance probabilities, ρ_a and ρ_d, are computed simply by finding the relative frequency of these events in the ground truth cell lineages of the training sequences. The appearance probability ρ_a accounts for both cell birth and cells moving into the image region, while the disappearance one ρ_d accounts for both cell death and cells moving out of the image. On the other hand, the migration and the division probabilities, ρ_{ij} and ρ_j, are obtained using a classification approach as described in the next section.

We define three sets of linear constraints to model cell behavior and exclude conflicting detection hypotheses from the solution, which we describe in the following.

Conservation of Flow: We require the sum of the flows incoming to a vertex to be equal to the sum of the outgoing flows. This allows for all the four cellular events while incurring their respective costs given in Eq. 6.

$$\sum_{e_{ij}\in E'} f_{ij} + f_{sj} + f_{dj} = \sum_{e_{jk}\in E'} f_{jk} + f_{jt}\ , \quad \forall j \in V'\ . \tag{7}$$

Prerequisite for Division: We allow division to take place at a vertex $j \in V'$ only if there is a cell at that location. We write this as

$$\sum_{e_{ij}\in E'} f_{ij} + f_{sj} \geq f_{dj}\ , \quad \forall j \in V'\ . \tag{8}$$

Exclusion of Conflicting Hypotheses: Given a hierarchy tree of detections, we define an exclusion set S_l for each terminal vertex $l \in V'$ of this tree. For instance, in the example of Fig. 2(d), the four exclusion sets are $\{a,b,d,g\}$, $\{a,b,d,h\}$, $\{a,b,e,i\}$ and $\{a,c,f,j\}$. Let S be the collection of all such sets for all the connected components in the sequence. We disallow more than one vertex from each set to appear in the solution, and express this as

$$\sum_{\substack{j\in S_l,\\ e_{ij}\in E'}} f_{ij} + \sum_{j\in S_l} f_{sj} \leq 1, \quad \forall S_l \in S\ . \tag{9}$$

We solve the resulting integer programs within an optimality tolerance of 0.001 using the branch-and-cut algorithm implemented in the Gurobi optimization library [52].

Note that our integer program contains only a single set of variables and three sets of linear constraints, and therefore it is more compact than those of the recent approaches of [2] and [1] that are similar to **ECLIP**. Formally, let N, M, and K denote the total number of detections, total number of exclusion sets and average number of neighbors per detection. The total number of constraints for **ECLIP** and [2] are $M + 2N$ and $M + 2N + 2NK$; the total number of variables are $N(3 + K)$ and $N(1 + K + 2^{(K+1)})$ respectively. The term $2^{(K+1)}$ comes from the fact that [2] includes indicator variables for higher order factors, which allows them to include local evidence for the total number of targets inside a merge-tree of segmentation hypotheses.

3.3 Cell Migration and Division Classifiers

Given two adjacent vertices $i, j \in V'$, and their associated ellipses e_i and e_j in consecutive time frames, we train a classifier to estimate the likelihood that both belong to the same cell. More specifically, we use a Gradient Boosted Tree (GBT) classifier [53] to learn a function $\varphi_{\text{migr}}(e_i, e_j) \in \mathbb{R}$, based on both appearance and geometry features including the distance between the ellipses, their eccentricities and degree of overlap, hierarchical fitting errors and ray features. Once $\varphi_{\text{migr}}(e_i, e_j)$ is learned, we apply Platt scaling to compute the probability ρ_{ij} that the two ellipses belong to the same cell, and plug it into Eq. 6.

Similarly, the likelihood of ellipse e_j at time t dividing into two ellipses e_k and e_l at $t + 1$ is learned with another GBT classifier, trained on features such as the orientation and size differences among the ellipses. We present a detailed list of both the migration and division features in the Supplementary Material.

For prediction, we compute the division score for ellipse e_j at time t as

$$\varphi_{\text{div}}(e_j) = \max_{\substack{e_{jk}\in E',\ e_{jl}\in E':\\ k\neq l}} \varphi_{\text{div}}(e_j, e_k, e_l)\ , \tag{10}$$

where $\varphi_{\text{div}}(\cdot,\cdot,\cdot)$ is the scoring function learned by the classifier, and (e_k, e_l) is a pair of ellipses corresponding to two potential daughter cells at time $t + 1$. We obtain the division probability ρ_j of Eq. 6 from $\varphi_{\text{div}}(e_j)$, again using Platt scaling.

4 Experiments

In this section, we first introduce the datasets and state-of-the-art methods we use as baselines for evaluation purposes. Our approach yields a statistically significant improvement over all baselines especially for the division and detection events. We achieve a p-value < 0.01 on the former, and a p-value < 0.005 on the latter using paired t-tests. Our software and video results can be downloaded from http://cvlab.epfl.ch/biomed/cell-tracking.

4.1 Test Sequences

We used the 10 training sequences with annotations from three datasets of the cell tracking challenge [5]. These sequences involve multiple cells that migrate, appear, disappear, and divide. Difficulties arise from low contrast to the background, complex cell morphology, and significant mutual overlap. We used the leave-one-out training and testing scheme within each dataset to train the classifiers of Section 3.3 and to learn the appearance and disappearance probabilities of Eq. 6.

- **HeLa Dataset:** It comprises two 92-frame sequences from the MitoCheck consortium. Cell divisions are frequent, which produces a dense population with severe occlusions.
- **SIM Dataset:** It comprises six 50- to 100-frame sequences. They simulate migrating and dividing nuclei on a flat surface.
- **GOWT Dataset:** It comprises two 92-frame sequences of mouse stem cells. Their appearance varies widely and some have low contrast against a noisy background.

4.2 Baselines

We compared our algorithm (**ECLIP**) against the following four state-of-the-art methods

- **Gaussian Mixture-based Tracker (GMM) [19]:** We ran the Gaussian Mixture Models approach of [19], originally designed to track cell nuclei, whose code is publicly available. We manually tuned its parameters to the ones that yield the highest tracking precision (TRA) [5] on each sequence.
- **KTH Cell Tracker (KTH) [3]:** The code is publicly available and has been reported to perform best in the Cell Tracking Challenge [5], [4]. We used the parameter settings optimized for each dataset and provided in the software package.
- **Conservation Tracking (CT) [1]:** We ran Ilastik V1.1.3 [54] that implements the method of [1]. We used the default parameters provided with the tool to handle appearance, disappearance, division and transition weights. The **CT** algorithm, like **ECLIP**, requires initial segmentations such as the ones shown in the second column of Fig.1. We used the same segmentations for both algorithms. We trained the division and the segment count classifiers separately for each dataset on manually labeled cells.
- **Joint Segmentation and Tracking (JST) [2]:** We ran the code of [2] that is publicly available. The model comprises several parameters for oversegmentation and tracking, which we tuned to achieve the highest possible TRA on each sequence.

It is worth noting that, in contrast to all the above trackers that require user-defined parameters, such as the weights of different components, our tracker treats all classifiers equally and thus does not require such parameters. As will be shown in Tab. 1, by doing so our tracker achieves the state-of-the-art results.

To demonstrate the importance of individual components of our approach, we also ran simplified versions of **ECLIP** with various features turned off:

- **Classifier Only (ECLIP-CL):** We threshold the output of our migration and division classifiers at a probability of 0.5 and return the resulting ellipse detections.
- **Best Hierarchy Only (ECLIP-BH):** For each ellipse hierarchy tree, we only keep the level that yields the minimum fitting error, which we define in the Supplementary Material. We then run our IP optimization on the resulting graphs, which are smaller than the ones we normally use.
- **Linear Programming Relaxation (ECLIP-LP):** We relax the integrality constraint on the variables and solve the optimization problem of Eq. 6 using linear programming. We then round the resulting fractional values to the nearest integer to obtain the final solution.
- **No Conflict Set Constraint (ECLIP-NC)**: We remove the conflict set constraints of Eq. 9 and solve the resulting integer program as before.
- **Fixed Division Cost (ECLIP-FD)**: We set the division probability to a constant p_d, which we compute by finding the relative frequency of the division event in the training sequences.

4.3 Evaluation Measures

We use precision, recall and the F-Measure, defined as the harmonic mean of precision and recall, to quantify the algorithms' ability to detect cell division, detection, and migration events. We also use two global measures, multiple object tracking accuracy (MOTA) [55] and tracking precision (TRA) [5], to evaluate the overall tracking performance. Note that for some datasets the ground-truth annotations are labeled dots rather than whole cell segmentation, such as HeLa. We therefore consider a cell to be detected if it overlaps the nearest annotation.

We follow the same evaluation methodology as in [1], which uses the connected components of the initial segmentations to compute the division, detection, and migration accuracies. We consider a cell migration event to be successfully detected if both connected components of the cell at t and $t+1$ are correctly identified. Similarly, we consider a division event to be successfully detected if it occurs at the correct time instant with the connected components of the parent and both daughter cells correctly determined. Finally, a detection event is said to be successfully identified if an algorithm infers the right number of cells within a connected component. Note that, our detection scores are computed using only the connected components that truly contain more than one cells. On the GOWT-2 sequence, which is different than the dataset C used in [1], the number of such connected components is small and CT produces a relatively large number of false positives, which explains the low precision.

Unlike the above measures, MOTA and TRA are defined on individual cell tracks rather than connected components that potentially contain multiple clumped cells. They therefore provide a global picture of tracking performance. TRA is an evaluation measure introduced for tracking cells by explicitly accounting for cell lineage and by imposing different penalties for different types of errors. MOTA, on the other hand, is an evaluation measure for generic multi-object tracking. To account for cell divisions, we have computed the MOTA score based on cell associations in two consecutive frames, where the associations can be either between two non-dividing cells or between one mother cell and one daughter cell. For a more thorough analysis of cell evaluation measures, please refer to [56].

		Division			Detection			Migration			MOTA	TRA	Time
		Rec	Pre.	F-M.	Rec.	Pre.	F-M.	Rec.	Pre.	F-M.			
HeLa-1	**GMM**	0.56	0.43	0.48	N/A	N/A	N/A	0.92	0.98	0.95	0.82	N/A	**44**
	KTH	0.65	0.72	0.68	N/A	N/A	N/A	0.95	**0.99**	0.97	0.91	**0.98**	70
	CT	0.74	**0.79**	0.77	0.56	**0.89**	0.69	0.94	**0.99**	0.97	N/A	N/A	74.85
	JST	0.79	0.55	0.65	N/A	N/A	N/A	0.86	0.92	0.89	0.73	0.80	128.16
	ECLIP	**0.92**	**0.79**	**0.85**	**0.96**	0.83	**0.89**	**0.97**	**0.99**	**0.98**	**0.94**	**0.98**	88.25
HeLa-2	**GMM**	0.40	0.18	0.24	N/A	N/A	N/A	0.95	0.98	**0.97**	0.43	N/A	**74**
	KTH	0.65	0.72	0.68	N/A	N/A	N/A	0.94	**0.99**	**0.97**	**0.90**	**0.97**	336
	CT	0.76	0.81	0.78	0.73	0.63	0.67	0.94	**0.99**	0.96	N/A	N/A	79.33
	JST	0.69	0.44	0.54	N/A	N/A	N/A	0.91	0.98	0.94	0.82	0.85	88.79
	ECLIP	**0.86**	**0.83**	**0.84**	**0.86**	**0.78**	**0.82**	**0.96**	**0.99**	**0.97**	**0.90**	**0.97**	232.31
GOWT-1	**GMM**	0.0	0.0	0.0	N/A	N/A	N/A	0.48	0.93	0.63	-0.29	N/A	39
	KTH	0.0	N/A	0.0	N/A	N/A	N/A	0.96	**1.0**	0.98	0.95	0.93	15
	CT	**0.50**	**1.0**	**0.67**	0.13	**1.0**	0.22	**0.98**	**1.0**	**0.99**	N/A	N/A	1.98
	JST	**0.50**	0.05	0.08	N/A	N/A	N/A	0.94	0.98	0.96	0.87	0.92	5.48
	ECLIP	**0.50**	**1.0**	**0.67**	**0.31**	**1.0**	**0.48**	**0.98**	**1.0**	**0.99**	**0.98**	**0.98**	**0.28**
GOWT-2	**GMM**	0.0	0.0	0.0	N/A	N/A	N/A	0.16	0.79	0.26	0.02	N/A	37
	KTH	0.0	N/A	0.0	N/A	N/A	N/A	0.94	**1.0**	0.97	0.94	0.91	16
	CT	**1.0**	0.17	0.29	**1.0**	0.02	0.03	0.95	**1.0**	0.97	N/A	N/A	0.81
	JST	**1.0**	0.02	0.04	N/A	N/A	N/A	0.93	0.98	0.95	0.85	**0.95**	3.87
	ECLIP	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**0.95**	**1.0**	**0.98**	**0.96**	**0.95**	**0.22**
SIM-1	**GMM**	0.0	0.0	0.0	N/A	N/A	N/A	0.65	0.91	0.76	0.45	N/A	14
	KTH	**1.0**	**1.0**	**1.0**	N/A	N/A	N/A	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	2
	CT	0.50	0.67	0.57	0.0	N/A	0.0	0.99	0.99	0.99	N/A	N/A	**0.20**
	JST	0.75	0.75	0.75	N/A	N/A	N/A	0.98	0.98	0.98	0.95	0.96	0.38
	ECLIP	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	0.99	**1.0**	0.99	**1.0**	1.16
SIM-2	**GMM**	0.0	0.0	0.0	N/A	N/A	N/A	0.76	0.90	0.83	0.56	N/A	15
	KTH	0.0	0.0	0.0	N/A	N/A	N/A	**1.0**	0.99	0.99	0.95	**1.0**	5
	CT	**1.0**	**1.0**	**1.0**	**1.0**	0.25	0.40	0.99	**1.0**	**1.0**	N/A	N/A	**0.41**
	JST	**1.0**	0.60	0.75	N/A	N/A	N/A	0.96	0.98	0.97	0.93	0.97	1.16
	ECLIP	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	0.75
SIM-3	**GMM**	0.0	0.0	0.0	N/A	N/A	N/A	0.77	0.94	0.85	0.66	N/A	20
	KTH	0.50	0.40	0.44	N/A	N/A	N/A	0.99	0.98	0.99	0.94	0.98	3
	CT	0.25	0.17	0.20	0.71	0.89	0.79	0.96	0.99	0.97	N/A	N/A	**0.39**
	JST	**1.0**	0.40	0.57	N/A	N/A	N/A	0.99	0.98	0.98	0.89	0.97	3.73
	ECLIP	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	1.00
SIM-4	**GMM**	0.25	0.33	0.29	N/A	N/A	N/A	0.91	0.94	0.92	0.81	N/A	23
	KTH	0.75	0.75	0.75	N/A	N/A	N/A	0.97	0.99	0.98	**0.96**	0.98	5
	CT	0.75	0.60	0.67	0.75	0.68	0.72	0.86	0.97	0.92	N/A	N/A	**1.69**
	JST	0.75	0.38	0.50	N/A	N/A	N/A	0.96	0.96	0.96	0.84	0.96	4.33
	ECLIP	**1.0**	**0.80**	**0.89**	**1.0**	**0.79**	**0.88**	**0.98**	**1.0**	**0.99**	**0.96**	**1.0**	1.85
SIM-5	**GMM**	0.0	0.0	0.0	N/A	N/A	N/A	0.77	0.87	0.82	-0.53	N/A	31
	KTH	0.0	0.0	0.0	N/A	N/A	N/A	**0.98**	0.99	0.98	0.96	**0.98**	10
	CT	0.50	0.33	0.40	0.93	0.82	0.88	0.96	0.99	0.98	N/A	N/A	8.94
	JST	0.50	0.13	0.21	N/A	N/A	N/A	0.94	0.97	0.96	0.88	0.97	**3.66**
	ECLIP	**0.75**	**0.75**	**0.75**	**1.0**	**1.0**	**1.0**	**0.98**	**1.0**	**0.99**	**0.98**	0.96	9.56
SIM-6	**GMM**	0.0	0.0	0.0	N/A	N/A	N/A	0.83	0.56	0.67	-0.49	N/A	27
	KTH	**1.0**	**1.0**	**1.0**	N/A	N/A	N/A	**0.99**	0.96	**0.97**	**0.95**	**0.99**	6
	CT	0.33	0.25	0.29	0.52	0.81	0.63	0.82	**0.99**	0.90	N/A	N/A	**1.28**
	JST	0.33	0.13	0.18	N/A	N/A	N/A	0.93	0.94	0.93	0.80	0.98	16.21
	ECLIP	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	0.95	**0.99**	**0.97**	**0.95**	**0.99**	5.82

TABLE 1: Comparison of our algorithm against state-of-the-art cell trackers in terms of tracking accuracy and running time. There are some numbers missing and shown as N/A's in the detection column. This is because the corresponding trackers do not output their intermediate segmentation results and therefore we are not able to compute the detection scores. The missing numbers in the TRA or MOTA column, on the other hand, are because those trackers do not produce complete cell tracks and thus we are not able to compute their TRA or MOTA.

4.4 Comparing against the Baselines

We ran our algorithm and the baselines discussed above on all the test sequences introduced in Section 4.1. Table 1 summarizes the results for a representative subset and the remainder can be found in the supplementary material.

Some numbers are missing because the publicly-available implementation of **CT** [1] we use does not provide the identities of individual cells in under-segmentation cases. We therefore cannot extract the complete tracks required to compute the MOTA and TRA scores. For our method and that of **CT**, we computed the detection scores by using the same segmentations obtained from the pixel classification approach of Section 3.1. By contrast, **GMM** [19], **KTH** [3] and **JST** [2] trackers take only raw images as input and do not accept external segmentations to be used. Therefore, we cannot compute their accuracy for the detection events. Finally, in some cases, **GMM** generates non-consecutive cell tracks, which is not accepted by the TRA evaluation software.

Table 1 shows that our tracker yields a significant improvement on the division and detection events, where we achieve p-values < 0.01 and < 0.005 respectively using paired t-tests. Even on the migration events for which the baselines already perform very well, we do slightly better. However, because the division and detection events are rare compared to migrations, the significant improvements on these two events only have a small impact on

		Division			**Detection**			**Migration**		
		Rec	Pre.	F-M.	Rec.	Pre.	F-M.	Rec.	Pre.	F-M.
HeLa-1	**ECLIP-CL**	0.95	0.06	0.11	N/A	N/A	N/A	0.98	0.47	0.64
	ECLIP-NC	0.48	0.14	0.22	0.0	0.0	0.0	0.96	0.93	0.94
	ECLIP-FD	0.78	0.81	0.80	0.96	0.85	0.90	0.97	0.99	0.98
	ECLIP-BH	0.88	0.73	0.80	0.81	0.87	0.84	0.97	0.99	0.98
	ECLIP-LP	0.92	0.80	0.86	0.95	0.81	0.87	0.97	0.99	0.98
	ECLIP	0.92	0.79	0.85	0.96	0.83	0.89	0.97	0.99	0.98
HeLa-2	**ECLIP-CL**	0.91	0.10	0.18	N/A	N/A	N/A	0.92	0.60	0.72
	ECLIP-NC	0.73	0.31	0.43	0.06	0.02	0.02	0.95	0.96	0.95
	ECLIP-FD	0.77	0.82	0.80	0.84	0.78	0.81	0.96	0.98	0.97
	ECLIP-BH	0.84	0.77	0.81	0.78	0.77	0.77	0.95	0.99	0.97
	ECLIP-LP	0.85	0.83	0.84	0.84	0.78	0.81	0.95	0.99	0.97
	ECLIP	0.86	0.83	0.84	0.86	0.78	0.82	0.96	0.99	0.97
GOWT-1	**ECLIP-CL**	0.0	0.0	0.0	N/A	N/A	N/A	0.98	0.89	0.94
	ECLIP-NC	0.50	0.50	0.50	1.0	0.29	0.44	0.98	1.0	0.99
	ECLIP-FD	0.50	1.0	0.67	0.31	1.0	0.48	0.98	1.0	0.99
	ECLIP-BH	0.50	1.0	0.67	0.25	1.0	0.40	0.96	1.0	0.98
	ECLIP-LP	0.50	1.0	0.67	0.31	1.0	0.48	0.98	1.0	0.99
	ECLIP	0.50	1.0	0.67	0.31	1.0	0.48	0.98	1.0	0.99
GOWT-2	**ECLIP-CL**	0.0	0.0	0.0	N/A	N/A	N/A	0.94	0.94	0.94
	ECLIP-NC	1.0	0.20	0.33	1.0	0.02	0.03	0.96	1.0	0.98
	ECLIP-FD	1.0	0.25	0.40	1.0	1.0	1.0	0.96	1.0	0.98
	ECLIP-BH	0.0	N/A	0.0	1.0	1.0	1.0	0.91	1.0	0.95
	ECLIP-LP	1.0	0.50	0.67	1.0	0.50	0.67	0.96	1.0	0.98
	ECLIP	1.0	1.0	1.0	1.0	1.0	1.0	0.96	1.0	0.98
SIM-1	**ECLIP-CL**	1.0	0.67	0.80	N/A	N/A	N/A	0.99	0.74	0.84
	ECLIP-NC	1.0	0.67	0.80	1.0	0.13	0.24	1.0	0.99	1.0
	ECLIP-FD	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.99	1.0
	ECLIP-BH	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.99	1.0
	ECLIP-LP	1.0	1.0	1.0	1.0	1.0	1.0	0.99	0.99	0.99
	ECLIP	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.99	1.0
SIM-2	**ECLIP-CL**	1.0	1.0	1.0	N/A	N/A	N/A	1.0	0.71	0.83
	ECLIP-NC	1.0	0.75	0.86	1.0	0.25	0.40	1.0	1.0	1.0
	ECLIP-FD	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
	ECLIP-BH	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
	ECLIP-LP	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
	ECLIP	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
SIM-3	**ECLIP-CL**	1.0	0.67	0.80	N/A	N/A	N/A	0.99	0.45	0.62
	ECLIP-NC	0.75	0.33	0.46	0.08	0.02	0.03	0.99	0.97	0.98
	ECLIP-FD	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
	ECLIP-BH	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.99	1.0
	ECLIP-LP	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
	ECLIP	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
SIM-4	**ECLIP-CL**	0.75	0.27	0.40	N/A	N/A	N/A	0.92	0.56	0.70
	ECLIP-NC	0.75	0.21	0.33	0.37	0.19	0.25	0.97	0.98	0.98
	ECLIP-FD	0.75	0.75	0.75	0.98	0.78	0.87	0.98	1.0	0.99
	ECLIP-BH	1.0	0.80	0.89	1.0	0.78	0.87	0.98	1.0	0.99
	ECLIP-LP	1.0	0.80	0.89	1.0	0.79	0.88	0.98	1.0	0.99
	ECLIP	1.0	0.80	0.89	1.0	0.79	0.88	0.98	1.0	0.99
SIM-5	**ECLIP-CL**	0.60	0.75	0.67	N/A	N/A	N/A	0.98	0.61	0.76
	ECLIP-NC	0.75	0.27	0.40	0.40	0.07	0.11	0.98	0.98	0.98
	ECLIP-FD	0.75	0.75	0.75	1.0	1.0	1.0	0.98	1.0	0.99
	ECLIP-BH	0.50	0.67	0.57	1.0	1.0	1.0	0.98	0.99	0.99
	ECLIP-LP	0.75	0.75	0.75	1.0	1.0	1.0	0.98	1.0	0.99
	ECLIP	0.75	0.75	0.75	1.0	1.0	1.0	0.98	1.0	0.99
SIM-6	**ECLIP-CL**	0.75	0.27	0.40	N/A	N/A	N/A	0.98	0.53	0.69
	ECLIP-NC	1.0	0.50	0.67	0.21	0.09	0.13	0.95	0.96	0.95
	ECLIP-FD	1.0	1.0	1.0	1.0	1.0	1.0	0.95	0.99	0.97
	ECLIP-BH	1.0	1.0	1.0	1.0	1.0	1.0	0.95	0.99	0.97
	ECLIP-LP	1.0	1.0	1.0	1.0	1.0	1.0	0.95	0.99	0.97
	ECLIP	1.0	1.0	1.0	1.0	1.0	1.0	0.95	0.99	0.97

TABLE 2: Tracking results with various features turned off. There are some N/A entries in the **ECLIP-CL** rows because it does not produce complete cell trajectories.

MOTA and TRA.

The comparatively poor performance of **GMM** can be partially ascribed to the fact that it relies on a simple hand-designed appearance and geometry model to detect individual cells. **KTH** relies on a richer appearance model that improves performance but requires tuning more parameters for each sequence. **CT** employs several cell-event classifiers and solves the tracking problem using integer programming, resulting in an overall higher division accuracy. Finally, the performance of **JST** is negatively impacted by the fact that it depends on a watershed-based oversegmentation that is sensitive to inaccuracies in pixel probability estimates. By contrast, our ellipse-fitting approach to generating competing hypotheses is robust to the ambiguous image evidence. Given the same initial segmentation as **CT**, our method achieves the best overall performance, thanks to the simultaneous detection and tracking.

Our tracker runs relatively fast even though we use a large number of variables. We show in Tab. 1 the running time of all trackers given the detections. In the SIM-4 case, the number of flow variables is around 1.1 million but the integer programming optimization takes only 2 seconds on a single Intel Xeon E5 core. In the HeLa-2 case, the number of variables is around 8 million and the optimization takes about 232 seconds. This is because in most cases, our GBT classifier produces a set of sparse and peaky probabilities for the cell events, which makes the convergence of the integer programming fast. Furthermore, our tracker is implemented in C++ using the state-of-the-art GUROBI library.

4.5 Evaluating Individual Components

To produce the results summarized by Table 1, we used our full approach as described in Section 3. In Table 2, we show what happens when we turn off some of its components to gauge their respective impacts.

ECLIP-CL relies on local classifier scores and does not impose temporal consistency. As a result, it produces many spurious cell tracks, which yields low precision for detection and migration events. **ECLIP-NC** addresses this by imposing temporal consistency but it allows multiple conflicting hypotheses to be active simultaneously. Therefore, it still suffers from spurious detections, which leads to low precision. **ECLIP-FD** disallows conflicting detections but relies on a fixed division probability, which is why it gives low division performance. **ECLIP-BH** uses division classifier costs but collapses the hierarchical dimension of our graphs and results in mis-detections. Finally, **ECLIP-LP** removes the integrality constraints on the flow variables. This gives a similar performance to **ECLIP** on most of the sequences suggesting that the integrality constraints are seldom helpful. However, in the case of HeLa-2, where division events are frequent, we observed that around 3% of the non-zero flow variables are fractional, which explains the 2% drop in recall compared to **ECLIP**.

5 Conclusion

In this work, we have introduced a novel approach to automatically detecting and tracking cell populations in time-lapse images. Our algorithm generates over-complete detection hypotheses based on fitting ellipses hierarchically, and compared to prior works [16], [2], it yields a much more comprehensible formulation of an integer program to choose the right detections from the over-complete set while jointly finding the optimal linking and division configuration. This results in more accurate trajectories and improved detection of mitosis events. Furthermore, our GBT classifiers are trained in a principled manner, and therefore our tracker can be readily applied for cell apoptosis caused by drug treatment.

In future work, we intend to extend our approach to tracking elliptical cells in $3D$ image sequences, as well as tracking cells of arbitrary shapes in both $2D$ or $3D$. In both cases, given the learned probabilities of different events, our network flow integer programming framework can be readily applied. However, to learn the event probabilities, we will have to change the features used to describe cells. In the case of elliptical cells in $3D$ sequences, we will need to fit ellipsoids to cells and train our classifier based on their features. To track cells of arbitrary shapes, we will have to develop new geometrcal features to capture their shapes. Given enough annotated training data, they should be obtainable using Convolutional Neural Networks. In the meantime, we are also interested in developing new evaluation measures that account for both cell tracking accuracy and sampling rate.

References

[1] M. Schiegg, P. Hanslovsky, B. Kausler, L. Hufnagel, and F. Hamprecht, “Conservation Tracking,” in *International Conference on Computer Vision*, December 2013, pp. 2928–2935.

[2] M. Schiegg, P. Hanslovsky, C. Haubold, U. Kothe, L. Hufnagel, and F. Hamprecht, “Graphical Model for Joint Segmentation and Tracking of Multiple Dividing Cells,” *Bioinformatics*, vol. 31, no. 6, pp. 948–956, 2015.

[3] K. Magnusson and J. Jalden, “A Batch Algorithm Using Iterative Application of the Viterbi Algorithm to Track Cells and Construct Cell Lineages,” in *International Symposium on Biomedical Imaging*, May 2012, pp. 382–385.

[4] M. Maška, V. Ulman, D. Svoboda, P. Matula, P. Matula, C. Ederra, A. Urbiola, T. España, S. Venkatesa, D. Balak *et al.*, “A Benchmark for Comparison of Cell Tracking Algorithms,” *Bioinformatics*, vol. 30, no. 11, pp. 1609–1617, 2014.

[5] C. O. Solorzano, M. Kozubek, E. Meijering, and A. M. noz Barrutia, “ISBI Cell Tracking Challenge,” 2014. [Online]. Available: http://www.codesolorzano.com/celltrackingchallenge/

[6] W. Ge and R. T. Collins, “Multi-Target Data Association by Tracklets with Unsupervised Parameter Estimation,” in *British Machine Vision Conference*, September 2008, pp. 93.1–93.10.

[7] A. Andriyenko, K. Schindler, and S. Roth, “Discrete-Continuous Optimization for Multi-Target Tracking,” in *Conference on Computer Vision and Pattern Recognition*, June 2012, pp. 1926–1933.

[8] A. Maksai, X. Wang, F. Fleuret, and P. Fua, “Globally Consistent Multi-People Tracking using Motion Patterns,” *arXiv preprint arXiv:1612.00604*, 2016.

[9] B. Yang and R. Nevatia, “An Online Learned CRF Model for Multi-Target Tracking,” in *Conference on Computer Vision and Pattern Recognition*, June 2012, pp. 2034–2041.

[10] C. Wojek, S. Walk, S. Roth, K. Schindler, and B. Schiele, “Monocular Visual Scene Understanding: Understanding Multi-Object Traffic Scenes,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 4, pp. 882–897, 2013.

[11] X. Wang, E. Turetken, F. Fleuret, and P. Fua, “Tracking Interacting Objects Using Intertwined Flows,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 11, pp. 2312–2326, 2016.

[12] K. Li, E. D. Miller, M. Chen, T. Kanade, L. E. Weiss, and P. G. Campbell, “Cell Population Tracking and Lineage Construction with Spatiotemporal Context,” *Medical Image Analysis*, vol. 12, no. 5, pp. 546–566, 2008.

[13] B. Kausler, M. Schiegg, B. Andres, M. Lindner, U. Koethe, H. Leitte, J. Wittbrodt, L. Hufnagel, and F. Hamprecht, “A Discrete Chain Graph Model for 3D+ T Cell Tracking with High Misdetection Robustness,” in *European Conference on Computer Vision*, October 2012, pp. 144–157.

[14] A. Vazquez-reina, M. Gelbart, D. Huang, J. Lichtman, E. Miller, and H. Pfister, “Segmentation Fusion for Connectomics,” in *International Conference on Computer Vision*, November 2011, pp. 177–184.

[15] J. Funke, D. Andres, F. A. Hamprecht, A. Cardona, and M. Cook, “Efficient Automatic 3D-Reconstruction of Branching Neurons from EM Data,” in *Conference on Computer Vision and Pattern Recognition*, June 2012, pp. 1004–1011.

[16] F. Jug, T. Pietzsch, D. Kainmüller, and G. Myers, “Tracking by Assignment Facilitates Data Curation,” in *MICCAI IMIC Workshop*, September 2014, pp. 1–12.

[17] D. Padfield, J. Rittscher, and B. Roysam, “Coupled Minimum-Cost Flow Cell Tracking,” in *Information Processing in Medical Imaging*, July 2009, pp. 374–385.

[18] ——, “Coupled Minimum-Cost Flow Cell Tracking for High-Throughput Quantitative Analysis,” *Medical Image Analysis*, vol. 15, no. 4, pp. 650–668, 2011.

[19] F. Amat, W. Lemon, D. Mossing, K. McDole, Y. Wan, K. Branson, E. Myers, and P. Keller, “Fast, Accurate Reconstruction of Cell Lineages from Large-Scale Fluorescence Microscopy Data,” *Nature Methods*, vol. 11, no. 9, pp. 951–958, 2014.

[20] K. Magnusson, J. Jalden, P. Gilbert, and H. Blau, “Global Linking of Cell Tracks Using the Viterbi Algorithm,” *IEEE Transactions on Medical Imaging*, vol. 34, no. 4, pp. 911–929, 2014.

[21] E. Meijering, O. Dzyubachyk, and I. Smal, “Methods for Cell and Particle Tracking,” *Methods in Enzymology*, vol. 504, no. 9, pp. 183–200, 2012.

[22] F. Li, Z. Yin, G. Jin, H. Zhao, and S. T. C. Wong, “Bioimage Informatics for Systems Pharmacology,” *PloS ONE*, vol. 9, no. 4, 2013.

[23] O. Dzyubachyk, W. V. Cappellen, J. Essers, W. Niessen, and E.Meijering, “Advanced Level-Set-Based Cell Tracking in Time-Lapse Fluorescence Microscopy,” *IEEE Transactions on Medical Imaging*, vol. 29, no. 3, pp. 852–867, 2010.

[24] A. Dufour, R. Thibeaux, E. Labruyere, N. Guillen, and J.-C. Olivo-Marin, “3D Active Meshes: Fast Discrete Deformable Models for Cell Tracking in 3D Time-Lapse Microscopy,” *IEEE Transactions on Image Processing*, vol. 20, no. 7, pp. 1925–1937, 2011.

[25] R. Delgado-gonzalo, N. Chenouard, and M. Unser, “Fast Parametric Snakes for 3D Microscopy,” in *International Symposium on Biomedical Imaging*, May 2012, pp. 852–855.

[26] M. Maška, O. Daněk, S. Garasa, A. Rouzaut, A. Munoz-barrutia, and C. O. de solorzano, “Segmentation and Shape Tracking of Whole Fluorescent Cells Based on the Chan-Vese Model,” *IEEE Transactions on Medical Imaging*, vol. 32, no. 6, pp. 995–1006, 2013.

[27] K. Zhang, L. Zhang, Q. Liu, D. Zhang, and M.-H. Yang, “Fast Tracking via Dense Spatio-Temporal Context Learning,” in *European Conference on Computer Vision*, September 2014, pp. 127–141.

[28] Z. Kalal, K. Mikolajczyk, and J. Matas, “Tracking-Learning-Detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 07, pp. 1409–1422, July 2012.

[29] S. Oron, A. Bar-hillel, and S. Avidan, “Extended Lucas-Kanade Tracking,” in *European Conference on Computer Vision*, September 2014, pp. 142–156.

[30] J. Zhang, S. Ma, and S. Sclaroff, “MEEM: Robust Tracking via Multiple Experts Using Entropy Minimization,” in *European Conference on Computer Vision*, September 2014, pp. 188–203.

[31] J. Gall, A. Yao, N. Razavi, L. Van Gool, and V. Lempitsky, “Hough Forests for Object Detection, Tracking, and Action Recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 11, pp. 2188–2202, 2011.

[32] H. Li, R. Sumner, and M. Pauly, “Global Correspondence Optimization for Non-Rigid Registration of Depth Scans,” in *Symposium on Geometry Processing*, July 2008, pp. 1421–1430.

[33] D. Padfield, J. Rittscher, N. Thomas, and B. Roysam, “Spatio-Temporal Cell Cycle Phase Analysis Using Level Sets and Fast Marching Methods,” *Medical Image Analysis*, vol. 13, no. 1, pp. 143–155, 2009.

[34] R. Collins and P. Carr, “Hybrid Stochastic / Deterministic Optimization for Tracking Sports Players and Pedestrians,” in *European Conference on Computer Vision*, September 2014, pp. 298–313.

[35] H. Jiang, S. Fels, and J. Little, “A Linear Programming Approach for Multiple Object Tracking,” in *Conference on Computer Vision and Pattern Recognition*, June 2007, pp. 1–8.

[36] X. Wang, V. Ablavsky, H. BenShitrit, and P. Fua, “Take Your Eyes Off the Ball: Improving Ball-Tracking by Focusing on Team Play,” *Computer Vision and Image Understanding*, vol. 119, pp. 102–115, 2014.

[37] A. Maksai, X. Wang, and P. Fua, “What Players Do with the Ball: A Physically Constrained Interaction Modeling,” in *Conference on Computer Vision and Pattern Recognition*, 2016.

[38] N. Chenouard and et al., “Objective Comparison of Particle Tracking Methods,” *Nature Methods*, vol. 11, no. 3, pp. 281–289, 2014.

[39] J. Berclaz, F. Fleuret, E. Türetken, and P. Fua, “Multiple Object Tracking Using K-Shortest Paths Optimization,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 11, pp. 1806–1819, 2011.

[40] A. Andriyenko and K. Schindler, “Globally Optimal Multi-Target Tracking on a Hexagonal Lattice,” in *European Conference on Computer Vision*, September 2010, pp. 466–479.

[41] H. Pirsiavash, D. Ramanan, and C. Fowlkes, “Globally-Optimal Greedy Algorithms for Tracking a Variable Number of Objects,” in *Conference on Computer Vision and Pattern Recognition*, June 2011, pp. 1201–1208.

[42] H. Heibel, B. Glocker, M. Groher, M. Pfister, and N. Navab, “Interventional Tool Tracking Using Discrete Optimization,” *IEEE Transactions on Medical Imaging*, vol. 32, no. 3, pp. 544–555, 2013.

[43] J. Kim, A. Bartoli, T. Collins, and R. Hartley, *Biomedical Image Registration*. Springer Berlin / Heidelberg, 2012, ch. Tracking by Detection for Interactive Image Augmentation in Laparoscopy, pp. 246–255.

[44] X. Wang, E. Turetken, F. Fleuret, and P. Fua, “Tracking Interacting Objects Optimally Using Integer Programming,” in *European Conference on Computer Vision*, September 2014.

[45] F. Li, X. Zhou, J. Ma, and S. T. C. Wong, “Multiple Nuclei Tracking Using Integer Programming for Quantitative Cancer Cell Cycle Analysis,” *IEEE Transactions on Medical Imaging*, vol. 29, no. 1, pp. 96–105, 2010.

[46] A. Merouane, N. Rey-Villamizar, Y. Lu, I. Liadi, G. Romain, J. Lu, H. Singh, L. Cooper, N. Varadarajan, and B. Roysam, “Automated Profiling of Individual Cellcell Interactions from High-throughput Time-lapse Imaging Microscopy in Nanowell Grids (TIMING),” *Bioinformatics*, vol. 31, no. 19, pp. 3189–3197, 2015.

[47] A. Khan, S. Gould, and M. Salzmann, “A Linear Chain Markov Model for Detection and Localization of Cells in Early Stage Embryo Developmenty,” in *IEEE Winter Conference on Applications of Computer Vision*, March 2014, pp. 526–533.

[48] A. Merouane, A. Narayanaswamy, and B. Roysam, *Video Bioinformatics*. Springer, 2015, ch. Integrated 5-D Cell Tracking and Linked Analytics in the FARSIGHT Open Source Toolkit, pp. 283–311.

[49] F. Jug, T. Pietzsch, D. Kainmller, J. Funke, M. Kaiser, E. van Nimwegen, C. Rother, and G. Myers, “Optimal Joint Segmentation and Tracking of Escherichia Coli in the Mother Machinen,” in *MICCAI BAMBI Workshop*, September 2014, pp. 25–36.

[50] J. Schindelin, I. Arganda-Carreras, E. Frise, V. Kaynig, M. Longair, T. Pietzsch, S. Preibisch, C. Rueden, S. Saalfeld, and B. Schmid, “Fiji: an open-source platform for biological-image analysis,” *Nature Methods*, vol. 9, no. 7, pp. 676–682, 2012, code available at http://pacific.mpi-cbg.de.

[51] D. K. Prasad and M. K. H. Leung and C. Quek, “ElliFit: An Unconstrained, Non-Iterative, Least Squares-based Geometric Ellipse Fitting Method,” *Pattern Recognition*, vol. 46, no. 5, pp. 1449–1465, 2013.

[52] Gurobi, “Gurobi Optimizer,” 2012, http://www.gurobi.com/.

[53] J. Friedman, “Stochastic Gradient Boosting,” *Computational Statistics & Data Analysis*, vol. 38, no. 4, pp. 367–378, 2002.

[54] C. Sommer, C. Straehle, U. Koethe, and F. Hamprecht, “ilastik: Interactive Learning and Segmentation Toolkit,” in *International Symposium on Biomedical Imaging*, March 2011, pp. 230–233.

[55] R. Kasturi, D. Goldgof, P. Soundararajan, V. Manohar, J. Garofolo, M. Boonstra, V. Korzhova, and J. Zhang, “Framework for Performance Evaluation of Face, Text, and Vehicle Detection and Tracking in Video: Data, Metrics, and Protocol,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 319–336, 2009.

[56] P. Matula, M. Maska, D. Sorokin, P. Matula, C. O. de Solrzano, and M. Kozubek, “Cell Tracking Accuracy Measurement Based on Comparison of Acyclic Oriented Graphs,” *PLoS ONE*, vol. 10, no. 12, 2015.