Tracking by Assignment Facilitates Data Curation

Florian Jug, Tobias Pietzsch, Dagmar Kainmüller, and Gene Myers

Max Planck Institute of Molecular Cell Biology and Genetics, Germany

Abstract Object tracking is essential for a multitude of biomedical research projects. Automated methods are desired in order to avoid impossible amounts of manual tracking efforts. However, automatically found solutions are not free of errors, and these errors again have to be identified and resolved manually. We propose six innovative ways for semi-automatic curation of automatically found tracking solutions. Respective user interactions are six simple operations: Inclusion and exclusion of objects and tracking decisions, specification of the number of objects, and one-click altering of object segmentations. We show how all proposed interactions can be elegantly incorporated into "assignment models" [1,2,3,4,5,6], an innovative and increasingly popular tracking paradigm. Given some user interaction, the tracking engine is capable of computing the respective globally optimal tracking solution efficiently, even benefitting from "warm start"-capabilities. We show that after interactively pointing at a single mistake, multiple segmentation and tracking errors can be fixed automatically in one single re-evaluation, provably leading to the new, feedback-conscious global optimum.

1 Introduction

Many research projects in molecular or developmental biology depend crucially on some form of object tracking. Famous examples are gene expression studies in cultured cells [7], nuclei tracking in *Drosophila* research [8], neuron tracing in connectomics [5], lineage tracking in early *C. elegans* [9] or *zebrafish* [4] embryos, and many more. Since the quality of subsequent biological conclusions in all these examples will depend on the obtained tracking results they must be as accurate as possible.

Today's predominant tracking methods originated in the 1960s [10,11] and were developed to track single or a hand-full of objects such as ships or airplanes. However, biological data often requires to track a large number of similar objects. In particular cases this can be resolved by maintaining multiple association hypotheses over multiple time-points [12]. However, although particle trackers and state space models can produce impressive results, proofreading is still required. Computer-assisted approaches for proofreading are usually not related to the automated method that produced the respective initial results.

Interactive error correction is rarely part of available tracking systems. Even more so they usually turn out to be difficult to implement and integrate, leaving

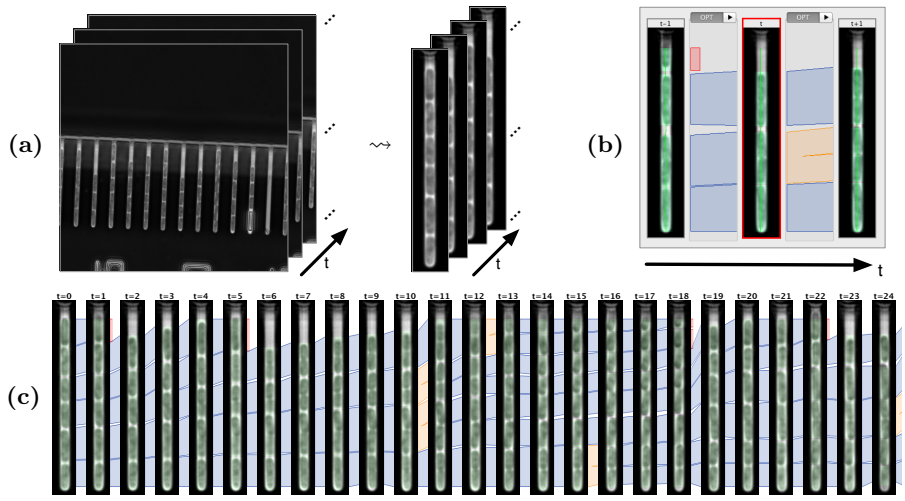

Figure 1: The Mother Machine data, the curation interface, and an example tracking solution. This tracking system was chosen to demonstrate the interaction capabilities of tracking systems knows as "assignment models" [1,2,3,4,5,6]. **(a)** Raw microscopy images and one preprocessed dataset showing only one of the growth channels containing cells to be tracked. **(b)** The user interface built to browse through tracking solutions. The goal is to resolve tracking errors with as little effort for users as possible. **(c)** The first 25 frames of an automatically found tracking solution. Cell mappings (blue), division assignments (yellow), and exit assignments (red) are shown between image frames.

the user with an inflexible patchwork of multiple tools they have to use. Reasons for this can be found in the way classical tracking models work. Their local iterative solvers are highly specialized, not offering intrinsic possibilities to constrain the space of possible solutions in any globally meaningful way. In other words they rarely offer any interaction capabilities that can be employed by users to prevent the system from making certain mistakes.

A relatively recent paradigm shift in available tracking methodology has the potential to change the game in all these respects – so called *Assignment Models*. The novelty of this type of tracking system is the way in which solutions are found. A tracking problem is formulated as an adequate global optimization problem which can be solved using discrete optimization methods.

In this article we introduce six simple, powerful user interactions for data curation: (*i*) Force solution to contain an object (segment), (*ii*) force solution to exclude segments, (*iii*) force tracking decision (assignment), (*iv*) avoid assignment, (*v*) specify object count in area, and (*vi*) alter segmentation. We will show that the very nature of the underlying optimization problem allows us to seamlessly incorporate these interactions.

In order to demonstrate the effectiveness of the proposed interactions we implemented them for an existing bacteria tracking system. We will present examples of how these interactions allow for simple and effective interactive curation of tracking solutions.

2 Background

Here we review a class of tracking methods called *assignment models* or *tracking by assignment*. We describe the underlying methodology using the example of an existing bacteria tracking system. We provide sufficient technical detail to prepare the reader for Section 3, where we will propose and implement several user interaction types that extend the existing system. However, it is important to note that none of the proposed interactions is specific to that particular system. Curation schemes, as we describe them, are directly applicable to other assignment models like [1,4,5,6].

Tracking by Assignment. Tracking consists of two equally important tasks: Cells need to be segmented in each frame, and segments of the same cell in different frames need to be linked. Tracking by assignment methods approach these tasks as a joint global optimization problem. In this context, the segmentation task is usually understood as selecting a subset of segments out of many possible hypotheses obtained from a (possibly heavy) oversegmentation of the input frames. Joint segmentation and tracking then boils down to enumerating many potential subsets of segments together with potential ways of linking (*assigning*) them between frames, and choosing from all these options a subset (solution) that is optimal in the following sense: Each of the potential segments and assignments is given a cost. Costs are simply a measure of how unlikely it is that this particular segment or assignment is part of the "real" solution to the tracking problem at hand [2,4] (also considering which other assignments are co-chosen). An optimal solution should minimize the summed costs.

Cost functions are usually designed to reflect the knowledge of domain experts. For example, if living cells are the objects to be tracked, the cost function for a cell division assignment that links one segment to two segments in the next frame is likely to contain a term that evaluates the size of the three segments to be linked. (Daughter cells should be similarly large and together be at least the size of the parent cell.)

Structural knowledge about which assignments can be chosen simultaneously is encoded in terms of constraints. They permit only physically meaningful solutions, i.e. solutions that do not describe impossible events like cells popping into existence out of nowhere, or cells moving to two places at once, etc. Constraints simply force or prohibit certain segments and assignments to be jointly active.

Given a segmentation and tracking problem formalized in terms of costs and constraints, well-established discrete optimization methods can be used to obtain a solution that is (*i*) feasible, i.e., free of conflicts, and (*ii*) cost-minimal. In the

following we will put these notions on formal grounds. We do this in the context of reviewing an existing tracking system [7].

Tracking Bacteria in the Mother Machine. The *Mother Machine* [13] is a microfluidic device designed to study living bacteria. It allows the observation and quantification of growth and division of the progeny of single "mother" cells over many generations using time lapse microscopy. Figure 1 shows sample data and a tracking solution found by the assignment model [7], reviewed below.

An excess of segment hypotheses is generated for each frame t, with many hypotheses partially overlapping and thereby providing alternative and mutually exclusive interpretations of parts of the image [7]. Each segment hypothesis as well as each possible assignment is represented as a binary variable v. In particular, each binary *segmentation variable* $h^{(t)} \in H^{(t)}$ corresponds to a particular segment hypothesis at frame t, while binary *assignment variables* $a^{(t)} \in A^{(t)}$ link segment hypotheses at time-point t to segment hypotheses at $t+1$. A mapping assignment $a^{(t)}_{i \mapsto j}$, for example, connects two segment hypotheses $h^{(t)}_i$ and $h^{(t+1)}_j$.

In any solution of the tracking problem a subset of the variables will be active ($v = 1$) indicating that the respective segment or assignment is part of the solution. As mentioned above, a cost function is defined that associates to every variable v the cost $c_v \in \mathbb{R}$ of including it in a solution. (For details on the cost function used for the mother machine we refer to [7]. In a nutshell, the cost measures how much a segment/assignment deviates from the expected appearance/dynamic behaviour of a bacteria cell.) The total cost C of a particular solution is then simply the summed cost over all active variables

$$C = \sum_i v_i \cdot c_{v_i}. \tag{1}$$

Linear constraints are used to restrict the solutions space to only include conflict-free and structurally sound solutions. As a simple example, two segment hypotheses which offer conflicting explanations of a particular pixel cannot simultaneously be active in any *feasible* solution.

We will look in detail at one particular constraint, thereby introducing some notation that will also be required later. *Continuity constraints* ensure that each active segment at frame t must be involved in exactly one assignment entering from frame $t-1$ and must also be involved in exactly one assignment towards $t+1$. In other words, a cell that has a past must have a future. Formally, this statement can be written as

$$\forall t \in \{2, \dots, T-1\},\ \forall h^{(t)} \in H^{(t)} : \sum_{a^{(t-1)} \in \Gamma_{\mathrm{L}}(h^{(t)})} a^{(t-1)} = h^{(t)} = \sum_{a^{(t)} \in \Gamma_{\mathrm{R}}(h^{(t)})} a^{(t)} \tag{2}$$

Here we use the *left neighborhood* $\Gamma_{\mathrm{L}}(h)$ and the *right neighborhood* $\Gamma_{\mathrm{R}}(h)$ of a segment variable h. The left neighborhood $\Gamma_{\mathrm{L}}(h)$ is the set of all assignments entering h from the previous frame. Similarly, the right neighborhood $\Gamma_{\mathrm{R}}(h)$ is the set of all assignments leaving h towards the next frame.

Finding the optimal solution using Integer Linear Programming (ILP). In tracking by assignment the globally optimal solution for picking a set of conflict free assignments of minimal cost is often found by solving an integer linear program (ILP) [4,5,6,7]. An ILP is an optimization problem that is fully specified by (*i*) an *objective function* that is a linear function of a set of variables $\mathcal{V}$, and (*ii*) a set of *constraints* that are formalized as (in-)equalities on these variables. The space of *feasible* solutions is defined by all variable assignments that obey all constraints. An *optimal* solution is a feasible solution that minimizes the objective function.

The joint segmentation and tracking formulation introduced above is already in ILP form: The set of variables $\mathcal{V}$ comprises binary segmentation and assignment variables. The objective to minimize is the cost C defined in Equation (1). (Note, that this is a linear function in the variables $v \in \mathcal{V}$.) In Equation (2) an example has been given of how constraints can be formalized as linear equalities.

Integer linear programming is a well-understood problem [14], and given the above formulation we can turn to off-the-shelf solvers like Gurobi™ to find an optimal solution. Although finding an optimal ILP solution is NP-hard, recent success solving relatively large tracking problems [4,5,6,7] suggests that assignment models pose well-natured instances to be solved as ILPs.

In the following we will make use of a particular feature of many ILP solvers, namely the ability to perform "warm-starts". One speaks about a warm start if a solver can benefits from residual intermediate results created during a preceding optimization. This can speed-up optimization significantly as shown in Figure 4.

The software presented in this paper was implemented in Java, using the imaging library ImgLib2 [15] and other components from the open source Fiji [16] universe. For solving ILPs we use Gurobi.

Reducing Redundancy by Substituting Segment Variables. The set of variables $\mathcal{V}$ contains variables h, for available segments, and a, for available assignments. Note that if and only if at least one assignment a that involves a segment i is active, also the the respective segment variable is active (i.e. $h_i = 1$). While it can be helpful to think in terms of dedicated segmentation variables, the model can drop them entirely [4,5,6,7]. After adequate constraints are added to the ILP each occurrence of $h_i^{(t)}$ can be substituted by a sum over all assignment variables in $\mathit{\Gamma}_{\mathrm{L}}(h_i^{(t)})$ (or $\mathit{\Gamma}_{\mathrm{R}}(h_i^{(t)})$).

3 Interaction Types and Implementation

In this section we show how the optimization problem used to solve an assignment model can be modified in response to user feedback. Interaction-based modifications and re-solving are iterated until the found solution reaches ground-truth quality. Our aim is to provide a curation interface for the bacteria tracking system discussed above by introducing several convenient user interactions. We will see that none of those interactions require significant changes to the existing tracking software.

The idea behind all proposed interaction types is simple: When a user identifies a segmentation or tracking error she suggests the correct alternative or simply points at the error, leaving the search for an alternative solution to the model. The given feedback is incorporated into the ILP via additional constraints or variables. Using warm-starts allows to solve the modified problem fast enough for interactive usage. Fixing a single error will usually resolve a bulk of follow-up errors as shown in Figures 2 and 3, and is quantified in Figure 4. What we offer is a curation system that can be used by non-expert users. Fixing an identified error usually requires not more than one mouse-click (or entry in a text-field).

In total we implemented six ways to interact with tracking solutions. While we show them in the context of the Mother Machine data, they can very well be implemented for other existing assignment models like [4,5,6]

IT1: Force Segment. A cell might be missing in the tracking solution, maybe even in multiple frames as shown in Figure 2(a). A user can hover the mouse over the part of the image where a cell was not picked up by the automated system, and respective inactive segment hypotheses are highlighted. Clicking on any highlighted segment will cause the system to (*i*) modify the ILP as described below, and (*ii*) re-run the solver to obtain the most probable (MAP) solution for the given data, now constrained to include the forced segment.

Technically this can be achieved by adding a single constraint to the ILP, namely $h = 1$ where h is the picked segment. Applying the redundancy reduction discussed at the end of Section 2, the constraint to be added can be expressed in terms of assignment variables as

$$\sum_{a \in \Gamma_{\mathrm{R}}(h)} a = 1, \tag{3}$$

where $\Gamma_{\mathrm{R}}(h)$ is the right neighborhood of h, i.e. the set of all assignments leaving h towards the next frame.

IT2: Exclude Segment. Analogously to letting the user pick missing segments, she can also decide to exclude segments from an erroneous solution (see Figure 2(b)). The re-solved ILP will correspond to the most probable (MAP) solution for the data, constraint to exclude the chosen segment. Analogously to IT1, the constraint to be added to the ILP is

$$\sum_{a \in \Gamma_{\mathrm{R}}(h)} a = 0. \tag{4}$$

IT3: Force Assignment. Instead of interacting with segments one might want to directly work with individual assignments. The user can browse through a library of available assignments. In Figure 2(c) a list of pre-filtered mapping assignments is shown. Any such assignment can be chosen to be included in future

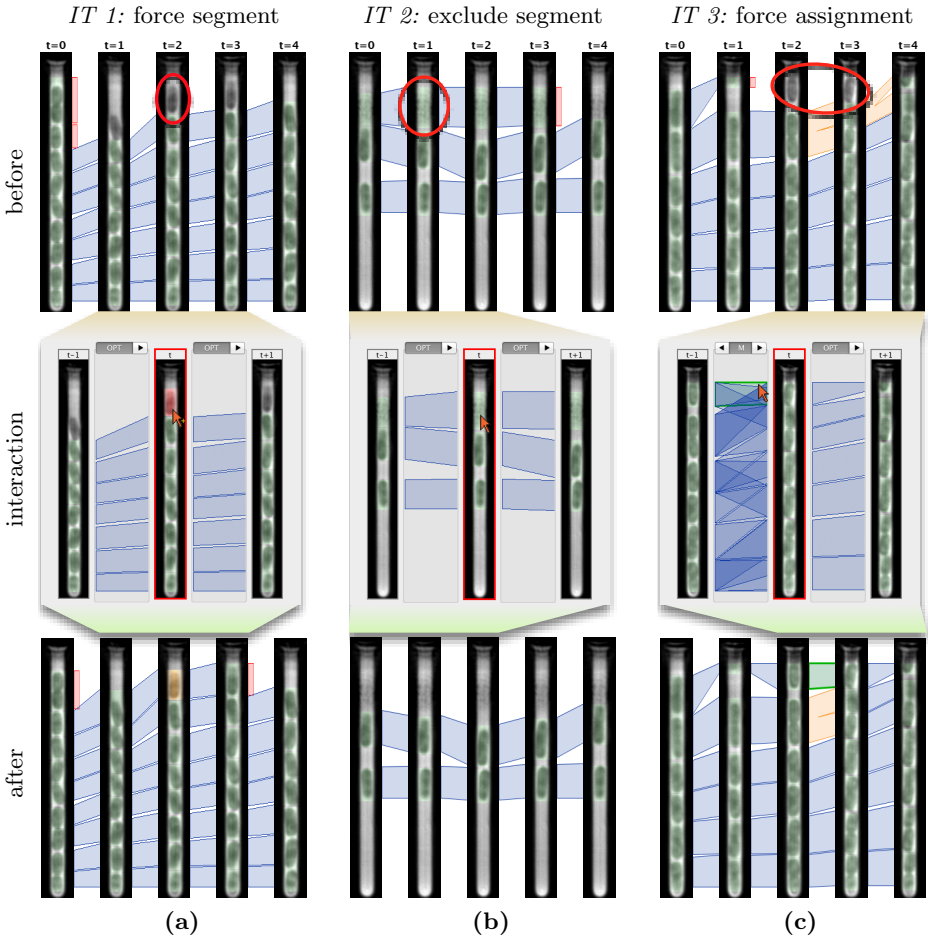

Figure 2: Interaction types (IT) 1 to 3. **(a-c)** Tracking errors (contained in top row) can be resolved in different ways (red cursors in middle row). The bottom row shows the resulting tracking solution after re-solving the model (for details see Section 2). Note that a single click by a user can cause multiple changes in the new, re-solved solution. Re-solving finds a globally optimal solution given the model and all user inputs. See also Figure 4 for a quantification of transitive changes caused by single user interactions. **(a)** The user can force a segment to be included in tracking solutions. Hovering over the a cell highlights the segments to be included. A single click causes the set of all solutions to be constraint to only the ones containing the selected segment. **(b)** By control-clicking on any pixel in a dataset, the user can exclude all segments that exist at that chosen pixel from the solution space. **(c)** Instead of interacting with segments, the user can directly select assignments to be included in curated solutions. A library of alternative assignments, here mapping assignments, are presented to the user. A choice is again made by a single click. (The chosen assignment is shown in green.)

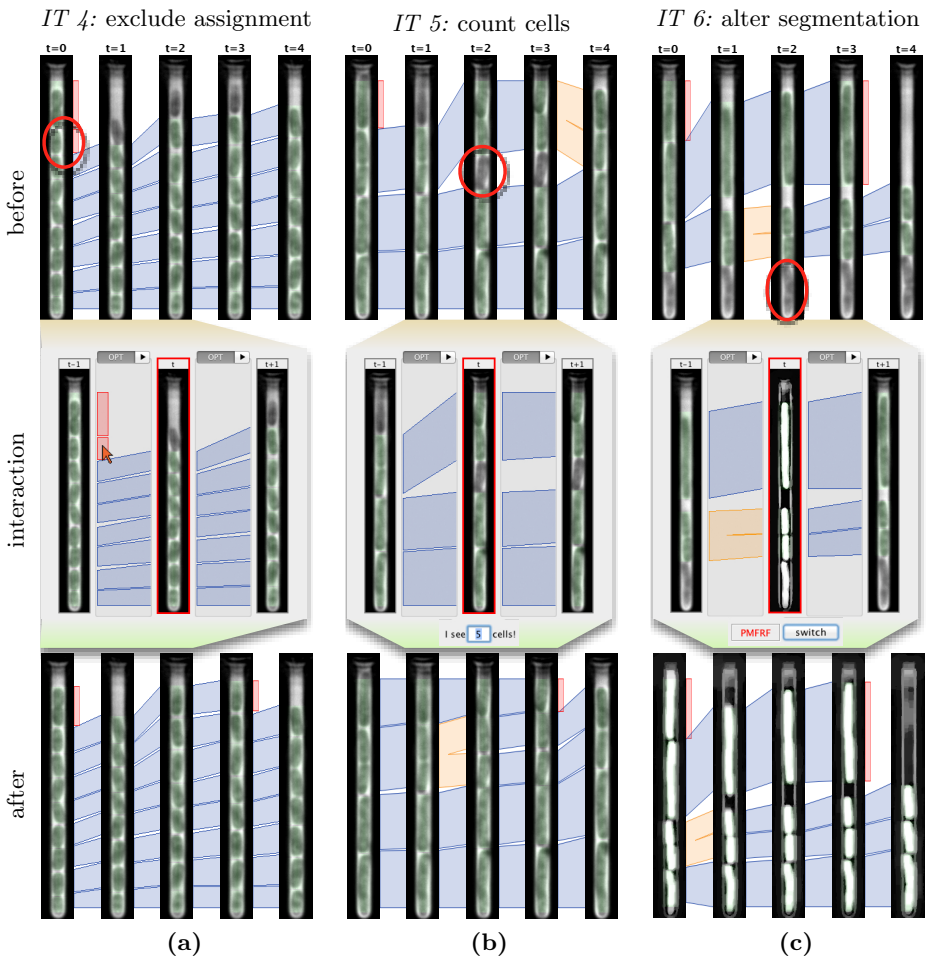

Figure 3: Interaction types (IT) 4 to 6. **(a)** Instead of interacting with individual segments, the user can directly exclude select assignments from the space of possible solutions. Simply clicking on an active assignment, here a faulty exit assignment, and re-solving (see Section 2) the model finds a new, globally optimal solution that complies to the model and all user interactions so far. **(b)** A very interesting, but slightly more "abstract" interaction type is shown here. The user can specify the number of cells in a given area, here in an entire time-point. By giving a cell count the search space of the model will be restricted to include only those solutions that contain the given number of activated segments in the corresponding frame. **(c)** If data quality is particularly bad in some parts of the images, a user can decide to replace segment hypotheses. The new ones, for example generated by a different, computationally more expensive segmentation method, can then be used in future model evaluations. Here we exchanged the segment hypotheses of all 5 shown frames and solved the modified model.

tracking solutions. Note that this also enforces the solution to contain specific segments, namely the segments that are associated by the forced assignment.

Browsing the library of available assignments can be done in only a few mouse-clicks. Please see the supplementary movie for a "live demo" of out interactive curation system.

Since there is precisely one binary variable a corresponding to the chosen assignment, the respective constraint to be added to the ILP to force this assignment to be 1 (on) is simply $a = 1$.

IT4: Exclude Assignment. Analogously to IT3 the user might also want to exclude an assignment that is part of the current tracking solution, as shown in Figure 3(a). For excluding an segment a single click is sufficient. The constraint to be added to the ILP for excluding an assignment is simply $a = 0$, where a is the chosen assignment.

IT5: Count Cells. For all previous interactions the user had to pick individual assignments or segments to include or exclude. Below we introduce an interesting interaction type that operates on a semantically higher level.

In Figure 3(b) we show a case where a cell was not included by the automatically found solution. But the user might at the same time be unsatisfied by the two cells on top of the growth channel to be segmented as a single object. The easiest way of to solve both problems at once would be to tell the model how ~~may~~ cells are contained in the current time-point. As before, let us constrain the solution space in the way we want – here ~~to~~ solutions that ~~containing~~ k segmented cells at time-point t. Formally this is the constraint

$$\sum_{h \in H^{(t)}} \sum_{v \in \Gamma_{\mathrm{R}}(h)} v = k, \tag{5}$$

where $H^{(t)}$ is the set of all possible segments at time t.

IT6: Alter Segmentation. So far, all interaction types depend on the existence of the correct segments in the pool of all segment hypotheses. This is of course not always the case. The last interaction we introduce aims at curating erroneous solutions by means of altering the segment pool.

Such alterations can be of different nature. Users could use the mouse or a stylus to draw an outline around an object, or use any number of interactive segmentation methods like [17] in order to segment the object of desire. We demonstrate a way to replace all segment hypotheses at a given time-point by segments created by an alternative segmentation method (which might be specialized to known weaknesses in parts of the data at hand).

For this type of interaction it is not enough to introduce a constraint alone. The modification of the ILP is of more fundamental nature since we have to remove some assignments (and associated variables v) from the ILP and replace them by new assignments between new and old segment hypotheses.

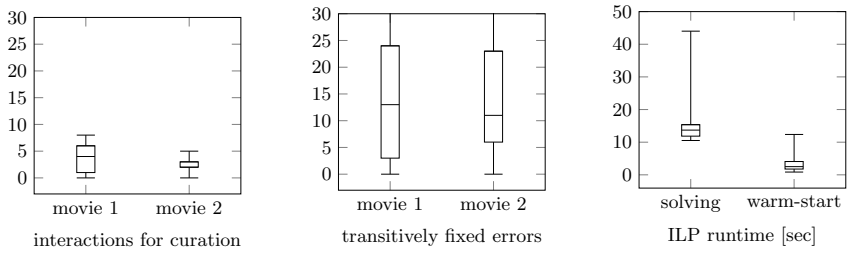

Figure 4: Erroneous segments and assignments and the number of interactions and CPU time needed to correct them. The left panel shows how many user interactions where needed to fully curate a dataset containing 100 frames. (Evaluated over a total of 21 (11 + 10) datasets from 2 time-lapse movies.) Middle panel shows how many segments and assignments had effectively been changed by the interactions in left panel. The right panel shows execution time (wall-time) for all datasets. We compare (*i*) initial runtime for setting up the model + solving it to global optimality, and (*ii*) warm-starts that where performed after each user interaction.

In Figure 3(c) we show how erroneous tracks can be fixed by switching the segmentation method at some (in the example five) frames and re-solving the modified ILP. Note that all necessary model changes are covered by procedures that were also needed to build the model in the first place.

4 Results

Figures 2 and 3 show proof-of-concept results for all six interaction types for the application of tracking bacteria in Mother Machine image data. For all these examples, median warm-start run-time is at 2.5 second (average 3.6), while a full run of the ILP solver without warm-start takes between 10 and 40 seconds. All times where measured on a MacBook Pro Retina (Fall 2012), see also Figure 4, rightmost panel.

We asked a biologist who was not involved in creating the proposed interactive curation system to use the interaction types of Figures 2 and 3 in order to fix all tracking errors contained in the automatically found solutions. After a short verbal introduction the user clicked through all errors of 21 all datasets available to us.

On average 3.05 interactions where needed to curate a 100 frame dataset to ground-truth quality [7]. For all 21 datasets 64 interactions lead to a grand total of 467 changed segments or assignments. This means that a single click or input by the user lead on average to 7.3 modifications in the final solution. This effect of *transitive error correction* is also visible in Figures 2 and 3. See also the first two box plots in Figure 4.

5 Discussion

We proposed six intuitive user interaction types that fit seamlessly into assignment models for joint segmentation and tracking. Clues given by the user are introduced into the problem formulation as additional constraints or variables. Given the revised model, the problem is immediately re-solved to optimality, making sure that the user is at no point left with an inconsistent or implausible solution. We showed that re-solving the bacteria tracking model fixes on average 7.3 segments and assignments at once. Owing to "warm-started" ILP re-solving the user interactions can be performed fast enough for avoiding annoying waiting times. (Re-solving time is usually below 3 seconds for datasets containing 100 time-points.)

The proposed interaction types are not specific to bacteria tracking. Other assignment tracking systems like [4,5,6], could easily be extended by very similar interactions. All six interaction types can be implemented by small changes to the underlying assignment models. In five of the six cases it is enough to add a single constraint to the ILP used to compute the global optimal tracking solution. But also IT6, which relied on changing segment hypotheses was implemented by calling existing code used to build the initial model.

The reason for the fast and easy realization of all 6 interactions is based on inherent features of assignment models – they are globally searching for optimal solutions and use solvers that operate on a clean, formal problem description that offers enough descriptive power to also capture user inputs (in form of constraints). As discussed elsewhere [8], many established tracking methods are less compliant to user interactions, possibly making assignment trackers the model of choice for interactive tracking systems.

Solving large ILPs might eventually become a bottleneck. In such a case one could drop the guarantee to find the global optimum by (*i*) switch to faster, approximate inference methods [18], or (*ii*) make use of adequate "divide-and-conquer" strategies like existing dual decomposition methods [19].

In conclusion we think that assignment models are a powerful new paradigm that can easily be made interactive to reach ground truth quality by including user feedback. Assignment models and interactions of the kind we presented here are arguably the best choice for interactive curation systems that biomedical imaging can offer today.

Acknowledgments. The authors want to thank David Richmond, Matthias Kaiser, and Erik van Nimwegen for mental support, Mother Machine data, and valuable feedback. We also want to thank Gaia Pigino for clicking through all available datasets. This work was supported by the German Federal Ministry of Research and Education (BMBF) under the funding code 031A099.

References

1. Jiang, H., Fels, S., Little, J.: A Linear Programming Approach for Multiple Object Tracking. CVPR (2007) 1–8
2. Padfield, D., Rittscher, J., Roysam, B.: Coupled Minimum-Cost Flow Cell Tracking. In: IPMI, Springer (2009)
3. Padfield, D., Rittscher, J., Roysam, B.: Coupled minimum-cost flow cell tracking for high-throughput quantitative analysis. Medical image analysis **15**(4) (2011) 650–668
4. Kausler, B., Schiegg, M., Andres, B., Lindner, M., Koethe, U., Leitte, H., Wittbrodt, J., Hufnagel, L., Hamprecht, F.: A Discrete Chain Graph Model for 3d+t Cell Tracking with High Misdetection Robustness. ECCV **7574** (2012) 144–157
5. Funke, J., Anders, B., Hamprecht, F., Cardona, A., Cook, M.: Efficient automatic 3D-reconstruction of branching neurons from EM data. In: CVPR, IEEE (2012)
6. Schiegg, M., Hanslovsky, P., Kausler, B., Hufnagel, L.: Conservation Tracking. ICCV (2013)
7. Jug, F., Pietzsch, T., Kainmüller, D., Funke, J., Kaiser, M., van Nimwegen, E., Rother, C., Myers, G.: Optimal Joint Segmentation and Tracking of Escherichia Coli in the Mother Machine. BAMBI/MICCAI (2014)
8. Jug, F., Pietzsch, T., Preibisch, S., Tomancak, P.: Bioimage informatics in the context of Drosophila research. Methods (2014)
9. Chen, L., Chan, L., Zhao, Z., Yan, H.: A novel cell nuclei segmentation method for 3D C. elegans embryonic time-lapse images. BMC bioinformatics **14**(1) (2013) 328
10. Kalman, R.: A New Approach to Linear Filtering and Prediction Problems. Journal of Basic Engineering **82**(1) (1960) 35–45
11. Bar-Shalom, Y.: Multitarget-multisensor tracking: Advanced applications. Norwood, MA, Artech House, 1990, 391 p. (1990)
12. Chenouard, N., Bloch, I., Olivo-Marin, J.: Multiple Hypothesis Tracking for Cluttered Biological Image Sequences. PAMI **35**(11) (2013) 2736–3750
13. Wang, P., Robert, L., Pelletier, J., Dang, W., Taddei, F., Wright, A., Jun, S.: Robust growth of E. coli. Current biology **20**(12) (2010) 1099–1103
14. Schrijver, A.: Theory of Linear and Integer Programming. J. Wiley & Sons (1998)
15. Pietzsch, T., Preibisch, S., Tomancak, P., Saalfeld, S.: ImgLib2–generic image processing in Java. Bioinformatics (2012)
16. Schindelin, J., Arganda-Carreras, I., Frise, E., Kaynig, V., Longair, M., Pietzsch, T., Preibisch, S., Rueden, C., Saalfeld, S., Schmid, B., Tinevez, J., White, D., Hartenstein, V., Eliceiri, K., Tomancak, P., Cardona, A.: Fiji: an open-source platform for biological-image analysis. Nature Methods (2012)
17. Rother, C., Kolmogorov, V., Blake, A.: Grabcut: Interactive foreground extraction using iterated graph cuts. In: SIGGRAPH, ACM (2004) 309–314
18. Kappes, J., Andres, B., Hamprecht, F., Schnörr, C., Nowozin, S., Batra, D., Kim, S., Kausler, B., Kröger, T., Lellmann, J., Komodakis, N., Savchynskyy, B., Rother, C.: A Comparative Study of Modern Inference Techniques for Structured Discrete Energy Minimization Problems. Preprint (2014) 1–32
19. Sontag, D., Globerson, A., Jaakkola, T.: Introduction to dual decomposition for inference. Optimization for Machine Learning (MIT Press) (2010)