PLOS COMPUTATIONAL BIOLOGY

RESEARCH ARTICLE

DeLTA 2.0: A deep learning pipeline for quantifying single-cell spatial and temporal dynamics

Owen M. O'Connor[1,2], **Razan N. Alnahhas**[1,2], **Jean-Baptiste Lugagne**[1,2]*, **Mary J. Dunlop**[1,2]*

1 Department of Biomedical Engineering, Boston University, Boston, Massachusetts, United States of America, **2** Biological Design Center, Boston University, Boston, Massachusetts, United States of America

* jlugagne@bu.edu (J-BL); mjdunlop@bu.edu (MJD)

Citation: O'Connor OM, Alnahhas RN, Lugagne J-B, Dunlop MJ (2022) DeLTA 2.0: A deep learning pipeline for quantifying single-cell spatial and temporal dynamics. PLoS Comput Biol 18(1): e1009797. https://doi.org/10.1371/journal.pcbi.1009797

Editor: Luis Pedro Coelho, Fudan University, CHINA

Received: August 10, 2021

Accepted: December 25, 2021

Published: January 18, 2022

Peer Review History: PLOS recognizes the benefits of transparency in the peer review process; therefore, we enable the publication of all of the content of peer review and author responses alongside final, published articles. The editorial history of this article is available here: https://doi.org/10.1371/journal.pcbi.1009797

Data Availability Statement: Code, installation instructions, and datasets are on Gitlab: https://gitlab.com/dunloplab/delta. Documentation for the software is available at: https://delta.readthedocs.

Abstract

Improvements in microscopy software and hardware have dramatically increased the pace of image acquisition, making analysis a major bottleneck in generating quantitative, single-cell data. Although tools for segmenting and tracking bacteria within time-lapse images exist, most require human input, are specialized to the experimental set up, or lack accuracy. Here, we introduce DeLTA 2.0, a purely Python workflow that can rapidly and accurately analyze images of single cells on two-dimensional surfaces to quantify gene expression and cell growth. The algorithm uses deep convolutional neural networks to extract single-cell information from time-lapse images, requiring no human input after training. DeLTA 2.0 retains all the functionality of the original version, which was optimized for bacteria growing in the mother machine microfluidic device, but extends results to two-dimensional growth environments. Two-dimensional environments represent an important class of data because they are more straightforward to implement experimentally, they offer the potential for studies using co-cultures of cells, and they can be used to quantify spatial effects and multi-generational phenomena. However, segmentation and tracking are significantly more challenging tasks in two-dimensions due to exponential increases in the number of cells. To showcase this new functionality, we analyze mixed populations of antibiotic resistant and susceptible cells, and also track pole age and growth rate across generations. In addition to the two-dimensional capabilities, we also introduce several major improvements to the code that increase accessibility, including the ability to accept many standard microscopy file formats as inputs and the introduction of a Google Colab notebook so users can try the software without installing the code on their local machine. DeLTA 2.0 is rapid, with run times of less than 10 minutes for complete movies with hundreds of cells, and is highly accurate, with error rates around 1%, making it a powerful tool for analyzing time-lapse microscopy data.

io/en/latest/. A Google Colab notebook, which allows users to test DeLTA 2.0 with their own data without installing the code on their local machine, is available at: https://colab.research.google.com/drive/1UL9oXmcJFRBAm0BMQy_DMKg4VHYGgtxZ.

Funding: This work was supported by the National Science Foundation (https://www.nsf.gov/) grants 2032357 and 1804096 and a subaward of National Institutes of Health (https://www.nih.gov/) grant AI153853 to MJD. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing interests: The authors have declared that no competing interests exist.

Author summary

Time-lapse microscopy can generate large image datasets which track single-cell properties like gene expression or growth rate over time. Deep learning tools are very useful for analyzing these data and can identify the location of cells and track their position. In this work, we introduce a new version of our Deep Learning for Time-lapse Analysis (DeLTA) software, which includes the ability to robustly segment and track bacteria that are growing in two dimensions, such as on agarose pads or within microfluidic environments. This capability is essential for experiments where spatial and positional effects are important, such as conditions with microbial co-cultures, cell-to-cell interactions, or spatial patterning. The software also tracks pole age and can be used to analyze replicative aging. These new features join other improvements, such as the ability to work directly with many common microscopy file formats. DeLTA 2.0 can reliably track hundreds of cells with low error rates, making it an ideal tool for high throughput analysis of microscopy data.

Introduction

The automation of hardware and software for microscopy has resulted in researchers' ability to generate massive datasets containing images of cells over time. For example, in a recent high throughput experiment Bakshi *et al*. imaged 10^8 *Escherichia coli* over days by acquiring 705 field of views every few minutes [1]. Additionally, recent studies have used closed-loop microscopy and optogenetic platforms to control gene expression in single cells in real time [2–4]. These improvements in microscopy have motivated the need for automated image analysis, as traditional approaches that require manual error correction cannot keep pace with the size of these new datasets or the rate at which they can be acquired. More generally, segmentation and tracking have historically required intensive user input as well as custom image processing code or experimental modifications such as the use of dedicated fluorophores [1,5–8]. These requirements limit throughput and can introduce burdensome experimental constraints.

To address this, researchers need tools that are rapid, accurate, and require minimal input from the user. This combination of needs is well suited for deep learning-based approaches, and deep convolutional neural networks have enabled fast and accurate analysis of images. Specifically, the U-Net architecture has emerged as the state-of-the-art convolutional neural network for biomedical applications [9]. U-Net uses a "U"-shaped network architecture with a contraction path, where successive convolutional layers are applied to feature maps that are progressively down-sampled, followed by a symmetric expansion path where the low-resolution but high-level encoding of the input image is up-sampled back to the original resolution. In addition, skip-connections are used to concatenate finer detail feature maps used in the contraction path with up-sampled feature maps in the expansion path. Skip-connections allow the network to retain high-resolution information needed to construct the mask at the end of the network. This approach has been widely successful for segmentation of cells [10–14] and for tracking cells from frame-to-frame within time-lapse images [11,13].

Here, we focus on analysis of bacterial time-lapse microscopy data in two-dimensional settings such as agarose pads or within microfluidic chips. With rapid cell cycle times, small cell sizes, and high throughput microfluidic devices, it is possible for researchers to generate large datasets containing thousands of single cells over periods of hours or days. As a result, researchers can use statistical analysis to study the subtle and complex effects of cell-to-cell

heterogeneity, gene expression dynamics, and cell-to-cell interactions in isogenic populations [1,6]. This has led to fundamental discoveries related to antibiotic resistance [8,15], and has allowed for accurate characterization of genetic parts and circuits [16]. Further, single-cell time-lapse analysis has revealed that cell division in *E. coli* is asymmetrical, where daughter cells receiving the 'old' pole grow more slowly than daughter cells receiving the 'new' pole [17]. However, these effects are subtle, necessitating measurements of many division events to determine statistically significant effects [18]. Until recently, such studies necessitated painstaking semi-manual analysis and curation of microscopy data. Software based on traditional image analysis techniques such as Schnitzcells [19], Oufti [20], and SuperSegger [21] all require significant user input and post-processing. A few recent studies have proposed deep learning models for bacterial cell segmentation such as MiSiC [10], DeepCell [22], and Cheetah [14], and yeast cell segmentation in Yeastnet [12] and Cell-DETR [23], however to our knowledge there is no integrated deep learning segmentation and tracking pipeline for two-dimensional time-lapse analysis of bacteria.

In previous work, we developed the Deep Learning for Time-lapse Analysis (DeLTA) pipeline to analyze single-cell growth and gene expression in microscopy images [11]. DeLTA uses two instances of the U-Net model to segment and then track cells. This allows for rapid and robust analysis of time-lapse movies. The original version of DeLTA focused on segmentation and tracking of cells in the 'mother machine' microfluidic device [24] where bacteria are constrained to narrow chambers where they grow in a single file line. This powerful design simplifies image analysis and enables experiments that run for many hours or days. However, this constrained geometry is not well suited for the study of two-dimensional effects such as diffusion of chemical signals, proximity-based effects, or co-cultures with mixed populations of cells. Two-dimensional configurations, ranging from microcolonies growing on agarose pads to microfluidic growth chambers, can be used to measure spatial dynamics of cell-to-cell interactions. Examples include quorum sensing [25] and the effect of efflux pumps on neighboring cells in the presence of antibiotics [26]. Segmenting and tracking cells in two dimensions are more challenging than for cells constrained within the mother machine. Segmentation becomes more difficult as microcolonies grow because images can contain hundreds of cells, where any given cell may have neighbors on all sides. The complexity associated with tracking also increases dramatically. In contrast to mother machine data, where frame-to-frame assignments can be limited to the small number of cells within the chamber (typically <10 cells), two-dimensional environments need to consider hundreds of possible assignments. Further, cells can move in any direction and may move large distances, for example if there is drift over the course of the movie.

In this manuscript we introduce DeLTA 2.0, a new version of DeLTA that segments and tracks cells in two dimensions. DeLTA 2.0 retains all the functionality of the original version and is fully compatible with mother machine data. In order to make our approach adaptable to different use cases, we minimized the number of pre- and post-processing steps so that most of the analysis is performed by the trainable models. The new version is available open source and uses a fully Python implementation. We have also introduced other improvements to the code to increase accessibility, such as the ability to work with images of arbitrary size and to accept many common microscopy file formats as inputs. We show that DeLTA 2.0 can segment and track co-cultures of bacteria growing on agarose pads and within microfluidic chambers. In addition to fluorescence and cell length, DeLTA 2.0 also records pole age. We use this to record replicative aging and compare the growth rate across generations. DeLTA 2.0 performs well on crowded images and requires no human intervention. The code, installation instructions, and datasets are available open source on Gitlab. We also provide a Google Colab notebook for users to rapidly test DeLTA 2.0 on their own data.

Results

DeLTA 2.0 can track cell length, growth rate, fluorescence, and progeny over time for cells growing in a two-dimensional microcolony. The algorithm takes microscopy images as inputs. It then uses two U-Net convolutional neural network models, one for segmentation and one for tracking, performs lineage reconstruction, and outputs single-cell data (Fig 1A). Segmentation generates information about cell morphology and can be used to identify both normal and filamented cells (Fig 1B and 1C). The tracking step reliably tracks cells across frames, recording division events when they occur (Fig 1D). Lineage reconstruction determines how

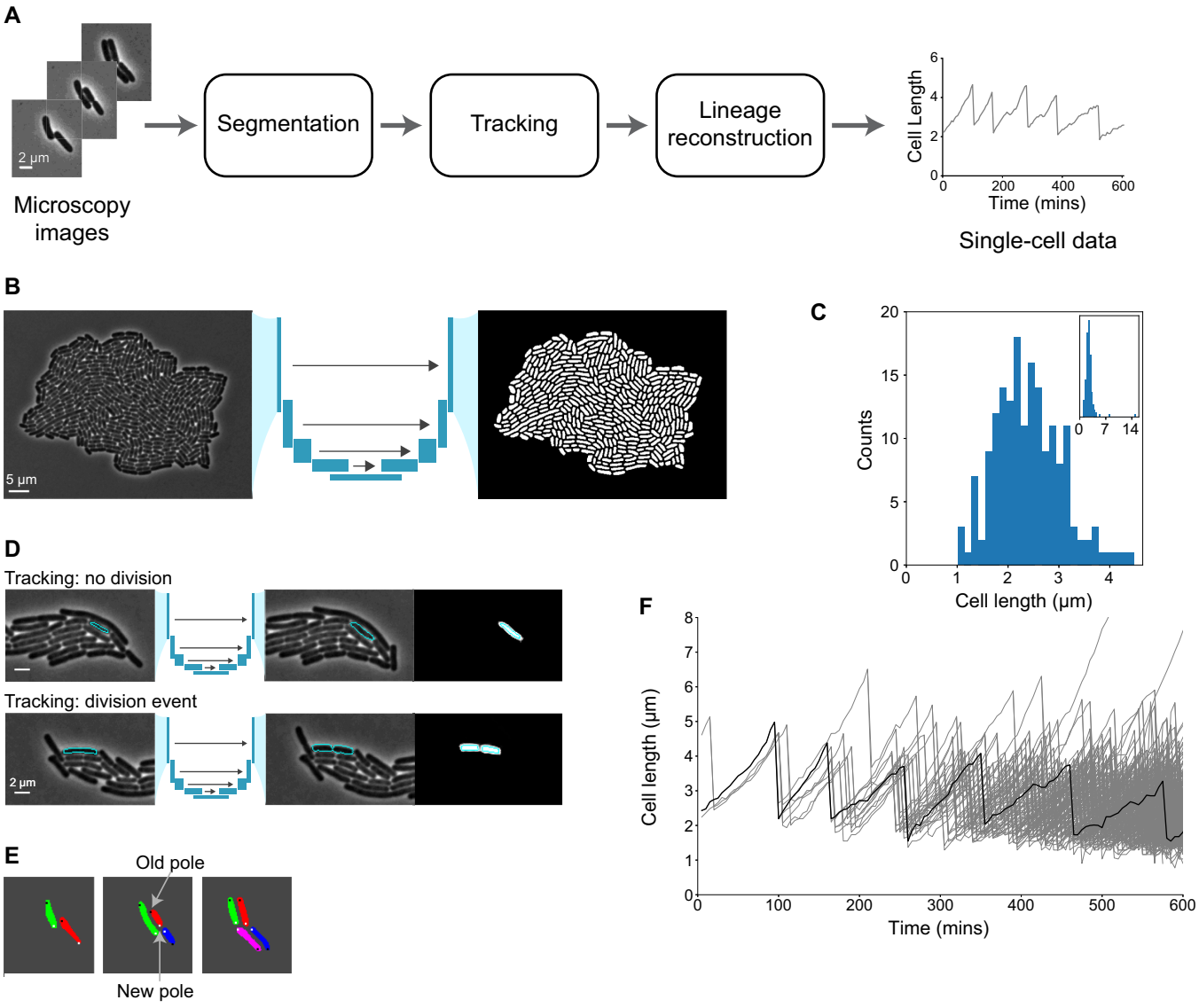

Fig 1. Segmentation and tracking of cells within a microcolony. (A) DeLTA pipeline consists of segmentation, tracking, and lineage reconstruction. **(B)** Segmentation example with phase contrast image containing an *E. coli* microcolony, which is input into a U-Net convolutional neural network to obtain segmentation results. **(C)** Histogram of cell lengths. Inset shows a zoomed-out version with outliers included. **(D)** Cell tracking between frames. Representative examples of cell tracking with and without division are shown with a phase contrast image of the 'previous frame' on the left, a phase contrast image of the 'current frame' in the middle, and a greyscale image of the 'prediction' on the right. The 'current frame' also shows the tracking prediction overlayed. The 'prediction' shows the U-Net output with the ground truth overlayed (S1 Fig). **(E)** Lineage reconstruction keeps track of cell lineages and records pole age. **(F)** Plot of cell lengths over time. Black line is a representative example of one cell's length as it grows and divides; all cells in the microcolony are shown in grey.

https://doi.org/10.1371/journal.pcbi.1009797.g001

cells are associated across generations and records features such as pole age (Fig 1E). The algorithm outputs single-cell resolution information for all cells within a field of view (Fig 1F). No human intervention is required to specify any input parameters. This is in sharp contrast to other methods, which typically require inputs, such as cell size or cell type [10,19–21,27].

DeLTA 2.0 can process datasets of various dimensions quickly and robustly. To evaluate its speed and accuracy, we used a movie from the literature that had been segmented and tracked with manual correction that DeLTA had not been trained on (Methods). It took 8 mins and 49 secs to conduct complete analysis of this time-lapse movie containing 69 frames, where the final frame contains 232 *E. coli* cells (S1 Movie). This analysis was conducted on a desktop computer with a Nvidia Quadro P4000 graphics card. In addition to being fast, DeLTA also has a low error rate. For the 3,286 cells segmented in the test set, there was a segmentation error rate of 0.01%. We defined a correct segmentation prediction as any case where the cell annotation in the model prediction had more than three quarters of its pixels overlapping with the ground truth data. This allows subtle differences between the prediction and ground truth to be considered acceptable, such as slight discrepancies in the exact location of the cell perimeter. In addition, if the model made a prediction where it erroneously connected two cells together, we defined this as two errors. Erroneously predicting a split cell was counted as one error. Because it can be difficult to assess the exact frame at which a cell divides, we did not count predictions where the model determined division events to be up to three frames later or earlier than in the ground truth as an error. Other metrics for assessing segmentation errors also showed good agreement with our findings that error rates are low (S2A Fig). The segmentation model tends to slightly under-segment due to emphasis on getting the cell borders classified correctly (S2B Fig). To assess the tracking error rate, we processed 6 movies of *E. coli* growing on agarose pads that the model had not been trained on. Out of the 17,622 tracking events, we measured an average error rate of 1.02%. We defined a correct prediction as a case where the cell assignment from one timepoint to the next matched the ground truth. Cases where the model assigned two cells as the daughters of a cell from the previous frame when there was in fact no division event, and cases where the model assigned no cell when the cell was in fact still within the field of view, were counted as tracking errors.

The DeLTA 2.0 algorithm has several improvements over the original version of DeLTA [11]. The new code is a purely Python workflow; movies do not need to be pre- and post-processed in Matlab. This transition allows the entire pipeline to exist in an open-source framework. We do provide code that can be used to convert the output to a Matlab file for users that are more comfortable working in this environment for post-processing data. In addition, in DeLTA 2.0 we take advantage of the Bio-Formats toolbox for Python [28,29]. This allows users to work directly with images in many common formats that are output via microscopy software, including nd2, czi, ome-tiff files, and many more, without the need for any preformatting. We also made updates to the code that increase its flexibility, while optimizing for performance. For example, DeLTA 2.0 can accept input images of various sizes. For large images (>512x512 pixels), DeLTA 2.0 will automatically crop the image into smaller windows for segmentation and then stitch the outputs back together. We note that large movies can cause memory issues, however the size limit at which this occurs will depend upon the configuration of the system on which it is run. For example, in our configuration the analysis of a time-lapse experiment with images of 1024x1024 pixels over 865 timepoints used 14GB of computer memory.

Since the original DeLTA code was optimized for images from the mother machine, where cells are constrained to one-dimensional chambers, tracking was relatively straightforward. In two dimensions, tracking is a more complex task and the number of cells that need to be tracked simultaneously increases dramatically. It can be challenging to identify which cells are

associated with which lineage. To improve tracking speed, we crop a 256x256 pixel area around the cell of interest. This approach works because a single cell is expected to remain in the immediate vicinity of where it was in the preceding frame, so it is reasonable to restrict the search for daughter cells to the local area. These coordinates are then used to crop the three other inputs (previous phase contrast image, current phase contrast image, and current segmentation). The dimensions of these cropping windows can be changed in the configuration file.

To reduce overfitting, DeLTA 2.0 uses several new data augmentation operations while training. In addition to operations such as random shifting, scaling, rotation, flipping, and illumination, which were present in the original software, we added three new functions, two for segmentation and one for tracking. To help deal with an occasional out of focus frame, we added a blurring function that slides a Gaussian kernel over the image (Methods). In addition, electronic noise is another issue when dealing with biological samples where the minimization of the total exposure to excitation light decreases the signal-to-noise ratio of the camera's sensor. To deal with this, we added a function that adds Gaussian noise (Methods). To simulate exaggerated cell movement during tracking, such as when an agarose pad dries out and causes the field of view to shift over time, we wrote a new augmentation function that introduces image translations between different timepoints (Methods). These operations help expand the training dataset and allow the model to generalize to realistic conditions.

Because drift of cells within images is a common concern for some applications, we further characterized the algorithm's performance under shifts of different sizes (S3 Fig). When the cell density is low, DeLTA can reliably handle shifts of up to ~30 pixels per timepoint in our images, which corresponds to ~4 μm. This problem is exacerbated in conditions where the frame is crowded with cells. When the cell density is high, performance begins to degrade after shifts of ~15 pixels, or ~2 μm. Thus, optimizing experimental conditions to minimize drift is important for high quality analysis.

To showcase the utility of DeLTA 2.0, we performed several experiments where we grew *E. coli* microcolonies on agarose pads and analyzed the output. First, we used DeLTA 2.0 to distinguish differences in growth rate between antibiotic resistant and susceptible cells grown in the same field of view. In this experiment, we mixed two strains of *E. coli*, one containing a tetracycline resistance gene and a constitutively expressed red fluorescent protein (RFP) reporter, and the other without the resistance gene and containing a green fluorescent protein (GFP) reporter. We grew cells in a co-culture on an agarose pad containing an inhibitory concentration of tetracycline (0.5 μg/ml). DeLTA 2.0 reliably segmented cells within the image (Fig 2A). The tetracycline resistance gene allowed the RFP-expressing cells to grow well whereas the tetracycline sensitive GFP-expressing cells grew very slowly (Fig 2B). The RFP and GFP fluorescence of individual cells can be plotted over time and shows the two distinct strains (Fig 2C). By extracting mean fluorescence levels for all cells within the time-lapse images, we found that fluorescence levels for the two populations were well-separated and maintained over time, as would be expected for the constitutive reporters. We also used DeLTA 2.0 to calculate the individual cell growth rates with respect to fluorescence. We observed two distinct clusters, corresponding to RFP cells that grew normally and GFP cells that grew slowly or did not grow (Fig 2D). These results highlight the ability to track cells with different properties simultaneously within the same movie.

DeLTA 2.0 is well suited for measuring growth and gene expression for many cells within an image. Because of this, another potential application is the study of replicative aging within bacterial microcolonies. Recently studies have shown that non-genetic differences may be passed down to offspring, causing a modest but measurable change in growth rate [7,17,18,30,31]. This can be tracked by recording pole age over time. Rod shaped bacteria have

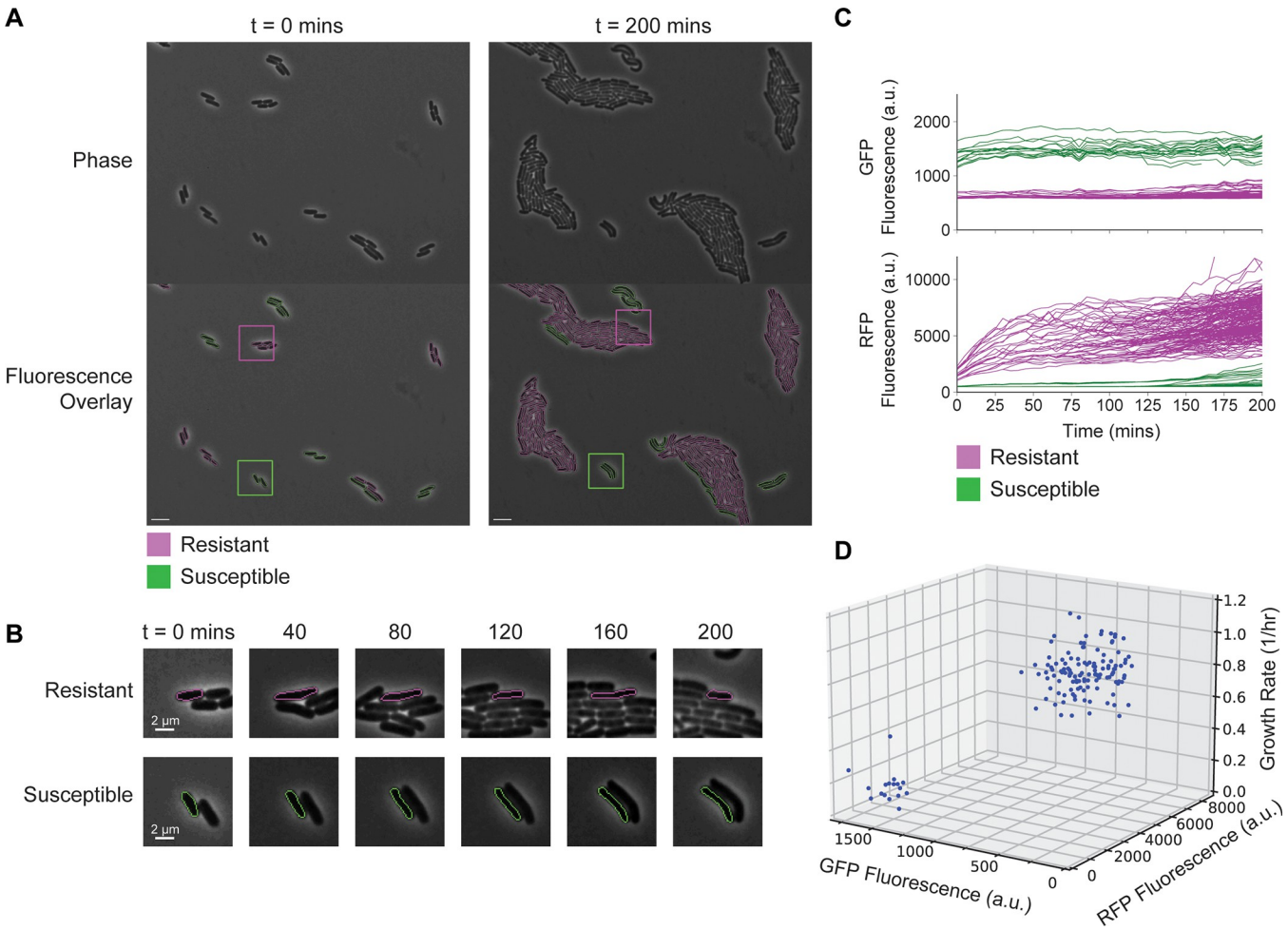

Fig 2. Resistant and susceptible strains of *E. coli* on agarose pads containing an inhibitory concentration of tetracycline. **(A)** Phase contrast images and associated fluorescence overlays. RFP expressing cells contain a tetracycline resistance gene and GFP expressing cells do not. The magenta and green cell outlines in the fluorescence overlay represent the resistant and susceptible cells, respectively. Region of interest boxes show the areas represented in (B). **(B)** Representative examples of antibiotic resistant and susceptible cells tracked over time. **(C)** RFP and GFP fluorescence tracked for individual cells over time. **(D)** GFP fluorescence versus RFP fluorescence for single cells plotted against growth rate. Fluorescence values are the averages over all the frames for that cell. For growth rate calculations, only cells that were present at t = 150 min were tracked, which is a time point mid-to-late in the movie. The analysis omits those cells that enter the field of view after t = 150 min since the growth rates become noisier with less data. Three resistant cell outliers with growth rates of ~1.4 1/hr are omitted from this view.

https://doi.org/10.1371/journal.pcbi.1009797.g002

two poles, where one end of the cell is referred to as the 'old' pole if it was passed down from the mother. The pole formed after division is referred to as the 'new' pole (Fig 3A).

To date, many experiments studying pole age have been conducted in the mother machine microfluidic device due to the ease of tracking cells [7,17,24]. However, a limitation of this approach is that it is only possible to track cells for a small number of generations because older generations are swept out of the imaging chamber while the mother cell's old pole stays at the dead end of the chamber. For this reason, the original DeLTA algorithm did not track pole age information, and in a division event the algorithm simply assigned the cell closest to the dead end of the chamber to be the mother and the other cell to be the daughter. However, on two-dimensional surfaces cells can be aligned in any orientation, therefore DeLTA 2.0 assigns old and new poles after division based on the position of the septum. To highlight the ability to track pole age over time, we analyzed a movie of *E. coli* growing in unstressed

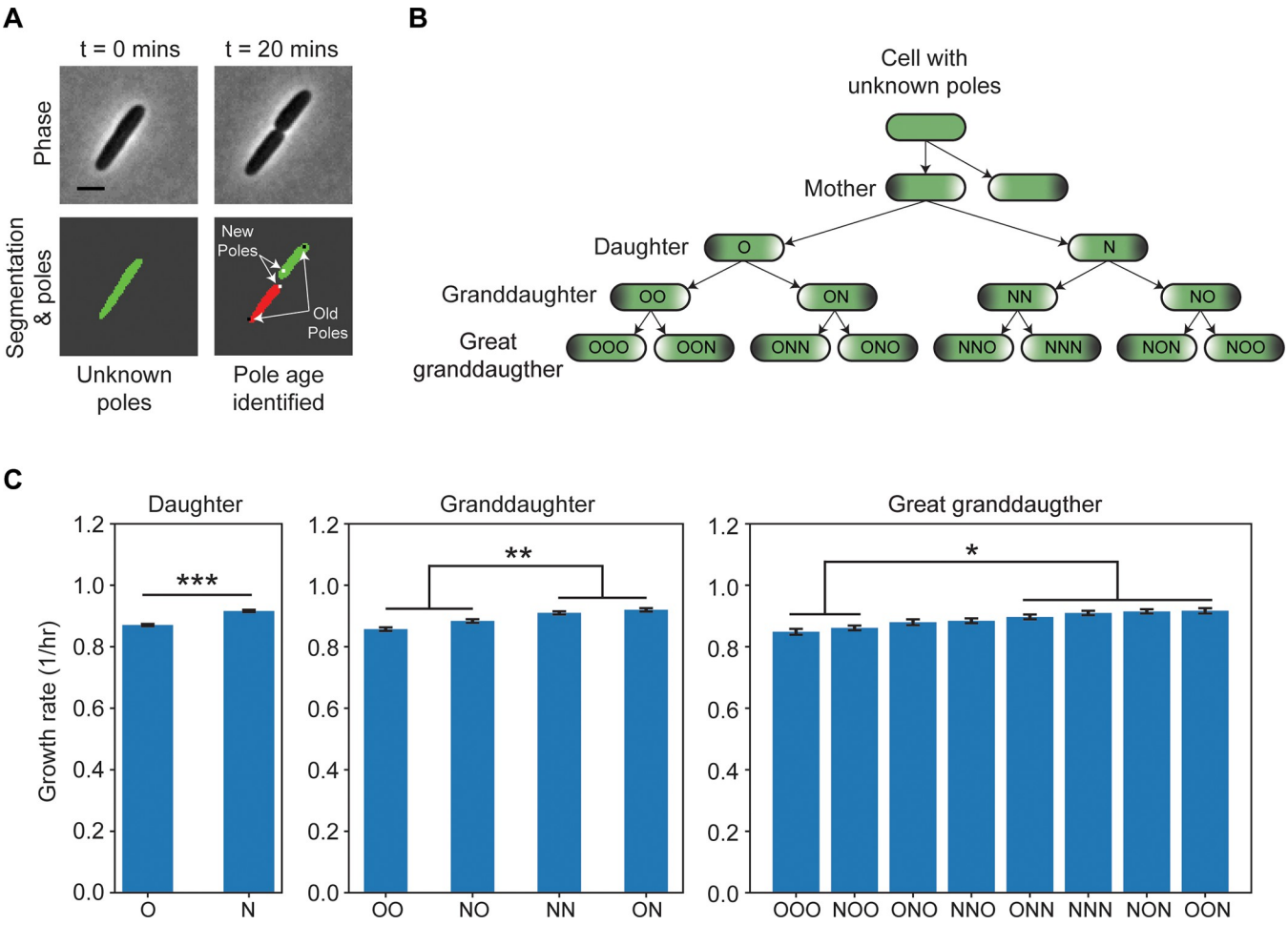

Fig 3. Pole age and its impact on growth rate. (A) Schematic showing how poles are passed down during a division. When a cell divides, the newly formed poles are defined as the 'new' poles (white dot) whereas the poles that were passed down from the mother are defined as the 'old' poles (black dot). Scale bar, 2 μm. **(B)** Pole assignment schematic. When the mother cell with known poles divides, the daughter cell that inherits the mother's old pole is denoted 'O' whereas the daughter that inherits the mother's new pole is 'N.' For each generation, either an O or an N is appended to the pole history. **(C)** Growth rate within each generation. The growth rate of an individual cell is calculated for the period right after the mother's division until right before the cell divides again. To reduce noise, only cells present for at least three frames were included in the analysis. Daughters (n = 11,246 cells; two tailed unpaired t-test; p-value $^{***} \leq$ 0.001), granddaughters (n = 10,726 cells; a one-way ANOVA with post hoc Tukey test used for statistical analysis. Statistical significance: 'OO' and 'NO' versus 'NN' and 'ON'; p-value $^{**} \leq 0.01$), and great granddaughters (n = 10,217 cells; a one-way ANOVA with post hoc Tukey test used for statistical analysis. Statistical significance: 'OOO' and 'NOO' versus 'ONN','NNN','NON', and 'OON'; p-value $^{*} \leq 0.05$). Error bars show standard error of the mean.

https://doi.org/10.1371/journal.pcbi.1009797.g003

conditions. At t = 0, we do not know the history of the cells, so the poles are initially unassigned (Fig 3B). After division, the mother's poles that are passed down become the 'old' poles and the newly divided poles are the 'new' poles of the daughters. This proceeds over subsequent generations. To keep track of this, we denote the daughter that receives the old pole as 'O' and the new pole as 'N'. When these daughter cells divide, they form cells that are granddaughters of the original mother cell. We append an O or N at the end of the pole sequence to record this. For example, the cell that inherits the original old pole is denoted OO and its sibling is ON. This continues through the generations, such that great granddaughters of the original mother have three letters in their pole sequence (e.g. OON).

First, we compared the growth rate of the old and new pole daughters from time-lapse movies of *E. coli* growing in unstressed conditions. Consistent with prior literature [17,18,30], we found that old pole (O) daughters grew more slowly than the new pole (N) daughters (Fig 3C).

Next, we used the rich generational information provided by DeLTA 2.0 to test for differences in growth rate between granddaughters and great granddaughters with different pole ages. We found that growth rate differences were dependent on which pole the cell most recently received. For instance, OO and NO had growth rates that were lower than the NN and ON (Fig 3C). Therefore, the old versus new pole influence upon growth rate is dominated by effects that extend back only one generation. This result was consistent with great granddaughters as well, where growth rates tended to be slower in cells that most recently received an old pole (OOO, NOO, ONO, NNO) than in those that received a new pole (ONN, NNN, NON, OON). These results demonstrate how pole age information can be tracked with DeLTA 2.0, enabling studies on replicative aging.

Finally, to test the generality of the algorithm we analyzed a time-lapse movie of >1,000 *B. subtilis* cells growing in a microfluidic device. This movie was generated in a different laboratory than any of the data that was included in the training set and no data from this movie were included in training. Using the previously trained model with no modifications, we analyzed the new movie and obtained excellent results (S2 Movie). The performance of DeLTA 2.0 on conditions it has not been trained for demonstrates its adaptability. Cropping to remove chamber edges and training to ignore device features such as support posts could further improve performance. This result highlights the broad potential for impact of DeLTA 2.0 in image analysis of two-dimensional bacterial cultures.

Discussion

In this work, we developed a deep learning pipeline that can process time-lapse images of bacterial microcolonies and output single-cell data. Utilizing the U-Net convolutional neural network architecture, the model can rapidly segment and track cells frame-to-frame with a low error rate. We applied this to successfully differentiate growth rates between sensitive and resistant strains of *E. coli* growing on an agarose pad. This demonstrates DeLTA's new ability to measure co-culture dynamics, which are hard to capture in devices like the mother machine. DeLTA 2.0 retains all the functionality of the original version of DeLTA and can now be used on mother machine data or microcolonies of bacteria growing in two dimensions.

The analysis pipeline is centered around the two trainable models for segmentation and tracking. We avoided hard-coding ad hoc rules such as pre- and post-processing steps or lineage reconstruction rules to the greatest extent possible. The current models work very well on standard rod-shaped bacteria such as *E. coli* (S1 Movie) and *B. subtilis* (S2 Movie). This suggests that analysis of cells with similar morphologies such as other *Bacillus*, *Pseudomonas*, and *Salmonella* species will be straightforward. DeLTA 2.0 may require further training for cases where the appearance of the cells deviates from the data the current models were trained on. For example, although the models can handle some elongated cells (Fig 1B and 1C), they are not currently optimized for cells with highly filamented morphologies or cells undergoing stress. However, since we avoided embedding rules in the code that are specific to our use case, we anticipate that DeLTA can be adapted not only to new morphologies after re-training but also to different organisms. For example, Fox *et al.* used DeLTA 1.0 to train a model to segment yeast cells with high accuracy [2].

Although segmentation works efficiently and has a low error rate, the model sometimes makes incorrect predictions about distinct cells being connected (S4 Fig). In particular, if movie frames are acquired at a high frequency, for example every minute or less, the segmentation model can fluctuate in its decision to split a cell undergoing division, which then complicates tracking. Within isolated images, these errors can be difficult to catch, even for humans. However, by looking at earlier and later frames it is usually possible to identify such errors

because cells cannot divide and then merge back together. As a potential future direction, deep learning architectures that use time-series information such as recurrent neural networks could be combined with our models to improve segmentation by incorporating temporal context.

The tracking model runs robustly but can slow down when there are hundreds of cells in the image. Because every cell in the frame creates an input for the tracking model, this increases exponentially as the bacterial microcolony grows. At present, for movies with many cells, tracking is the current bottleneck for processing, whereas segmentation is comparatively faster. For example, S2 Movie, which has 100 frames and over 1000 cells in each frame takes a total time of ~1 hour to process with our system configuration, with over 50 minutes attributed to calculating the tracking events. Future efforts to optimize the tracking algorithm could help to address this by avoiding methods that scale linearly with the number of cells. In addition, initial tests suggest that it may be possible to decrease the size of the convolutional neural network by removing layers, though this would likely need to be customized for specific applications (S5 Fig). This could be applied to the segmentation or tracking model.

Overall, DeLTA can now process two-dimensional movies accurately and capture spatial dynamics in a high throughput manner with no human intervention. It works with many common microscopy file formats and extracts single-cell features such as cell poles, length, lineage, and fluorescence levels automatically and saves data into Python and Matlab compatible formats. As many microbiology researchers work with these types of data, we envision that this software can be used to increase the throughput of microscopy image analysis.

Methods

Implementation and network architecture

Code, installation instructions, and datasets are on Gitlab: https://gitlab.com/dunloplab/delta. Documentation for the software is available at: https://delta.readthedocs.io/en/latest/. We also provide a Google Colab notebook, which allows users to test DeLTA 2.0 with their own data without installing the code on their local machine:

https://colab.research.google.com/drive/1UL9oXmcJFRBAm0BMQy_DMKg4VHYGgtxZ.

The U-Nets are implemented in TensorFlow/Keras. We developed a fully-integrated pipeline that can compile single-cell data from Bio-Formats compatible files or TIFF image sequences, but we also provide simpler scripts and data that illustrate the core principles of our algorithm for easy adaptation to different use cases. In S1 Table, we have listed the packages and exact versions that we use in our environment to run DeLTA 2.0. Both the segmentation and tracking models implement a U-Net neural network architecture. The tracking model inputs a phase contrast image and segmentation from the current and previous frame and outputs a greyscale image of the predicted tracking event (S1 Fig). In the original version of DeLTA [11], the tracking model uses a softmax function as the final activation layer and a categorical cross-entropy loss function to produce three greyscale output images with 1's in each layer representing the mother cell, daughter cell, and background, respectively. In DeLTA 2.0, the tracking model uses a sigmoid function as the final activation layer and a pixelwise-weighted binary cross-entropy loss function to produce a single greyscale output image with 1's representing tracked cells (mother and potential daughter) and 0's representing the background and the cells that did not track to the input cell.

Loss functions and training

To train both models, we implemented a pixelwise-weighted binary cross-entropy loss function, as in the original U-Net paper [9]. This loss function was adapted from the binary cross-

entropy function in Tensorflow/Keras which measures the pixelwise loss of a sample. The loss is multiplied elementwise with the weight map to magnify or reduce the loss (S6 Fig). Lastly, we normalized the loss based on the sum of the total weight map to evenly distribute how much each sample updates the model. Overall, the loss function determines the difference between the model output and the ground truth, which is then used to update the weights within the model. As in Ronneberger *et al.* [9], our loss function takes a weight map as an extra input to assign more importance to certain pixels in the ground truth during training. We used custom weight maps to improve segmentation on rod-shaped bacteria by increasing weights for the center of the cells and the borders between the cells. We also minimize weight on the background, where background is defined as anything in the image that is not a cell or border (S7 Fig). More specifically, we maximized the weights for the skeletons of the cells and borders, which are pixel-wide representations of binary objects in images [32]. Determining the exact borders of the cells by eye is hard and partially arbitrary. To prevent the model from learning these arbitrary cell-border interfaces, we reduced the weights in these areas (S7 Fig). In addition, the background weights were set to be variable, where the weight increased with respect to an incorrect prediction (S8 Fig). The model outputs a number between 0 and 1, with 0 representing the background and 1 representing a cell. Since we had high confidence in the ground truths for the background, we were able to set the values of the weight map for the background to be equal to the actual prediction for the background. If the model incorrectly predicted a cell for a pixel that is background, then there would be a high value for that pixel in the weight map. Alternatively, if the model correctly predicted background for a pixel that was background, then there would be a low value for that pixel in the weight map. This method allows the model to efficiently recognize and discard debris and reduce overfitting on the background. Our code includes a function to automatically generate these weight maps from the ground truth segmentations. Custom weight maps were also implemented for the tracking model, although they were found to be less critical to training a successful model. The weight map was similarly generated by applying morphological operations to the segmentation of all cells in the current frame and the ground truth. The skeleton of the ground truth cell was set to the highest weight while other cells' pixels were set to decreasing weights depending on the distance from the tracked cell (S1B Fig). The function generating these maps is also provided in our software.

In addition to the data augmentation operations described in the original version of DeLTA [11], two new data augmentations were used while training the segmentation model. We used the blurring function GaussianBlur from the OpenCV package which convolves a 5x5 Gaussian kernel over the image. For the noise function we used the random.normal function from the numpy package which outputs random samples from a Gaussian distribution into an array the same size as the image. This is added to the original image and rescaled back to a range between 0 and 1. Both the blurring and noise functions are applied to all input images with user-specified standard deviations. We set 1 and 0.03 to be the default standard deviations for the blurring and noise functions, respectively.

To simulate exaggerated cell movement during tracking, such as when an agarose pad dries out and causes the field of view to shift over time, we added a function that randomly shifts the inputs containing the current frame (microscopy image of the current frame, segmentation mask of all the cells in the current frame, ground truth, and weight map) up to a user-specified number of pixels (e.g. 5 pixels). These operations help expand the training dataset and allow the model to generalize to realistic conditions.

The segmentation model used to quantify the error rate was trained for 600 epochs with 300 steps per epoch and a batch size of 1. The Adam optimizer was used with a learning rate of 10^{-4}. The tracking model used to quantify the error rate was trained for 500 epochs with 300 steps per epoch and a batch size of 2. The Adam optimizer was used with a learning rate of

10^{-5}. In all cases, the models converged during these training period. For example, S9 Fig shows convergence results for segmentation model training.

Training set generation and testing

For the segmentation training dataset, the initial segmentations were generated semi-automatically by an expert using the interactive learning and segmentation toolkit Ilastik [33]. This accounted for 11% of the final training set. Once the segmentation model was performing well on the test data, which we defined as being more than 95% accurate, we used it to generate more training data. Incorrect DeLTA 2.0 outputs, like segmentations that connect two distinct cells together, were manually corrected. Processed DeLTA outputs accounted for 36% of the training set. Additionally, we incorporated published segmentation data from van Vliet *et al.* [34] where cells were segmented and tracked to measure the spatial dynamics of gene expression in bacterial microcolonies. We obtained the cell segmentation and tracking data from the ETH archive: https://doi.org/10.5905/ethz-1007-77. On this dataset, we performed operations to improve the data quality including smoothing filters, dilation, erosion, and skeletonize functions. These data accounted for the remaining 53% of the training set. The final training set had 307 training examples from sixty movies, with a maximum of 10 frames per movie to increase sample diversity. Each training example consisted of a phase contrast image as the input, the corresponding segmented ground truth, and a pre-generated weight map used in the loss function (S7 Fig). The test movie used to evaluate the segmentation model was colony 150310–05 from the trpL data zip file from van Vliet *et al.* [34] on the ETH archive.

For the tracking training dataset, we used a modified version of the Matlab script used in DeLTA 1.0 to generate the initial training examples. Instead of showing the whole frame in the graphical user interface, the modified script showed a zoomed-in 75x75 pixel box around the cell of interest. In addition, the modified script had one output consisting of the mother and daughter cell whereas the original script had three outputs for the mother cell, daughter cell, and background. The Matlab script was used to produce 15% of the training set. Each training example consisted of four inputs, one output, and one weight map. The inputs were the phase contrast of the previous frame, segmentation of the cell of interest in the previous frame, phase contrast of the current frame, and segmentation of all the cells in the current frame (S1A Fig). The output is a segmentation mask for the cell(s) that the cell of interest in the previous frame tracked to. The weight map is used in the loss function. Once the tracking model was performing with more than 99% accuracy on test data, we used it to generate more training data. Movies with images taken 5 minutes apart were processed using DeLTA 2.0 and then new training examples were generated by tracking cells across longer time intervals. Instead of tracking from the frame immediately before the current timepoint, cells were tracked from a frame from two or three timepoints before. This allowed us to generalize to longer acquisition intervals and to situations where the cells grew faster or travelled further between frames. These processed DeLTA outputs accounted for 20% of the training set. In addition, published tracking data from van Vliet *et al.* was incorporated to increase the training set size, accounting for 65% of all the training examples. The final tracking training set had 23,655 examples. The test movies used to evaluate the tracking model were colony 140408–01 from the cib data zip file; colonies 151029_E1-1, 151029_E1-5, and 151101_E3-12 from the rpsM data zip file; and colonies 150309–04 and 150310–05 from the trpL data zip file from van Vliet *et al.* [34].

Time-lapse microscopy experiments

Overnight cultures of *E. coli* MG1655 were diluted 1:100 and allowed to grow for 1–2 hours in LB medium. For the co-culture experiment, we included 30 μg/mL of kanamycin for plasmid

maintenance. We created the co-culture with a 1:5 dilution by mixing 0.5 μL of the resistant strain and 0.5 μL of the susceptible strain with 4 μL of LB medium. For the pole age experiment, the culture was diluted 1:100 in MGC media (M9 salts supplemented with 2 mM MgSO4, 0.2% glycerol, 0.01% casamino acids, 0.15 μg/ml biotin, and 1.5 μM thiamine). A 1:100 dilution was used to decrease cell density. For both experiments, 1–1.5 μL of the diluted samples were added to prewarmed 1.5% low melting temperature agarose pads made with MGC media. Samples were prepared and imaged as described in Young, *et al.* [19]. A Nikon Ti-E microscope was used with a 100x oil objective for all microscopy experiments.

In the co-culture experiment, *E. coli* MG1655 were transformed with a single plasmid, either with tetracycline resistance or without. Both plasmids originated from the BioBrick plasmid library (pBbA7k) [35]. The plasmid for the resistant cells harbors both a tetracycline resistant gene and red fluorescent protein gene (*rfp*) (pBbA7k-RFP-tetA), while the sensitive cells contain only green fluorescent protein (*gfp*) (pBbA7k-sfGFP). The pads contained 30 μg/mL kanamycin for plasmid maintenance and 0.5 μg/mL tetracycline. Phase contrast, GFP, and RFP fluorescence images were taken every 5 minutes.

In the pole age experiment, *E. coli* MG1655 was used and no antibiotics were present in the culture. Phase contrast images were taken every 5 minutes.

We calculated the growth rate as:

$$Growth\ Rate = \frac{L_t - L_{t-1}}{L_{t-1}} \times \frac{1}{\Delta t}$$

Where L_t is the cell length at time t, L_{t-1} is the length at time t-1, and Δt is the difference in time between t and t-1 (5 min = 0.083 hr in our movies). Growth rates were measured for one generation. For example, to measure the growth rate of OO, measurements start when O divides into OO and end when OO divides into OON and OOO. However, no information about the growth rate of O is used to calculate the growth rate of OO. The growth rate is the average across the timepoints within this generation. To reduce noise, growth rates were only recorded in the analysis if cells were present for at least three frames.

Testing impact of frame shifts on tracking

We considered three representative conditions and performed tracking under artificial frame shifts (S3 Fig). The conditions were cases of "low," "medium," and "high" cell density, which we defined as having <5, 20–25, and >200 cells in the field of view. We calculated the percent overlap between the model prediction and ground truth for all tracking events as the frame was artificially shifted. For each tracking event, the field of view was shifted over 11 different distances ranging from 1 to 100 pixels, which correspond to 0.129 to 12.9 μm, and each distance was applied in all four cardinal directions to generate a total of 48 shifts plus one unshifted version. The time-lapse movie used for this analysis was colony 151101_E3-12 from the rpsM zip file from van Vliet *et al.* [6].

Supporting information

S1 Fig. Schematic showing how the tracking model uses the inputs to make predictions. **(A)** The four inputs are concatenated into one array and processed by the model, which outputs a greyscale image of the cell tracked (or cells, in the case of a division event). **(B)** Ground truth and custom weight map for tracking.
(EPS)

S2 Fig. Accuracy of the segmentation model. (A) For each frame in the movie we plotted the intersection-over-union (IOU) / Jaccard index, Dice score / F1 coefficient. Error rate, as defined in the manuscript, is also shown. **(B)** The proportion of pixels that were True Positives (TP) + True Negatives (TN), False Positives (FP), and False Negatives (FN) for each frame in the movie. Note that these are pixelwise predictions as opposed to predictions per cell. TP and TN represent the rate of correct pixelwise predictions in the frame. FP represent the rate of erroneous predictions of pixels as part of a cell when the ground truth reports it as background. FN represent the rate of erroneous predictions of pixels as background. The test set (S1 Movie) was used to calculate these evaluation metrics.
(EPS)

S3 Fig. Impact of frame shifts on tracking model performance. The average prediction in the four cardinal directions to ground truth overlap for 20 representative tracking events plotted as a function of the shift distance. Tracking model performance when **(A)** cell density is low (<5 cells in the frame), **(B)** medium (20–25 cells in the frame), or **(C)** high (>200 cells in the frame). As the shift distance increases, the performance decreases. The model generally performs better with fewer cells to track per frame. The black line represents the mean for each shift distance.
(EPS)

S4 Fig. Limitations of segmentation. Two sequential phase contrast images of *E. coli* microcolonies with their respective segmentations. The red arrow points to an error where the model has incorrectly combined two cells into one. This type of error is very hard to correct out of context.
(EPS)

S5 Fig. Reducing the size of the model to increase speed. Schematics showing different network architectures. **(A)** Original U-Net architecture that we use throughout the paper. **(B)** U-Net architecture without the bottom layer. **(C)** U-Net architecture without the bottom two layers. The model has been trained on segmentation as well as tracking for these reduced networks. The network in (C) runs twice as fast as the original network in (A) and sacrifices very little accuracy.
(EPS)

S6 Fig. Schematic of the pixelwise binary cross-entropy loss function used to train the segmentation and tracking models. The inputs and outputs for the loss function are shown. The inputs include the ground truth (GT), the prediction made by the model (Pred), and the associated weight maps. The output is the pixelwise-weighted loss, which is a greyscale image. In step 1, the ground truth and prediction are used to calculate the binary cross-entropy loss. The first half of the equation measures the pixelwise loss associated with the model predicting background when the ground truth is a cell. The second half of the equation measures the pixelwise loss associated with the model predicting cell when the ground truth is background. In step 2, the weighted loss is calculated by performing an elementwise multiplication of the weight map and the loss calculated in step 1. The weight map helps the model learn the more important features, such as cell borders. This schematic of the loss function is simplified for visualization purposes. For clarity, we also show how the pixelwise-weighted loss maps onto the prediction, where cyan regions highlight areas where the loss is emphasized in order to improve the model's performance for these regions of the example image.
(EPS)

S7 Fig. Training the model on segmentation with custom weight maps. Schematic showing two models trained on the same dataset with different weight maps. In this example, both U-Net models were trained for 600 epochs, 300 steps per epoch, with a batch size of 2. Inputs necessary to train the model include the phase contrast image, the associated segmented ground truth, and weight map. **(A)** Model trained with weight maps derived from the original U-Net paper. Green ovals show examples of errors. **(B)** Model trained with custom weight maps which were generated by applying morphological operations to the segmented ground truth. The model trained on the new weight maps performs better, as shown by the outputs on a test image. **(C)** To aid visualization, the 'Overlay' shows the custom weight map overlayed on the phase contrast image. The overlay shows that the weights are emphasized at the core of the cells (shown by red lines) and at the borders (shown by yellow lines).
(EPS)

S8 Fig. Utilizing variable background weight maps for training the model on debris. A simplified schematic showing how the loss is calculated for a single input using weight maps. **(A)** Schematic showing the traditional use of weight maps. **(B-C)** Schematics showing the use of variable weight maps. The prediction is used to update the weight map. **(B)** The background weight map values are replaced by the background values in the prediction. This method forces the model to quickly learn to filter out debris as the weight map values for the background increase significantly when the model predicts debris as cells. **(C)** Conversely, when the model correctly classifies the debris as background, the weight map values for the background remain similar to the original values.
(EPS)

S9 Fig. Loss history of segmentation model trained over 600 epochs. Loss of the model during training as a function of the total number of epochs. Only points where the model performance improves and loss reaches a new minimum are shown. There were no improvements in the last 85 epochs, showing convergence.
(EPS)

S1 Movie. Time-lapse movie of a bacterial microcolony analyzed with DeLTA 2.0. Phase contrast images containing *E. coli* cells outlined with different colors representing unique cells. Cells can be tracked by following their respective colors throughout the movie. White arrows indicate cell division events. White and colored dots refer to the new and old poles, respectively. This time-lapse movie was part of the test set used to calculate the error rate for tracking and segmentation. Timestamp shows time in HH:MM format.
(MP4)

S2 Movie. Time-lapse movie of a dense bacterial microcolony growing in a microfluidic device analyzed by DeLTA 2.0. Phase contrast images show *B. subtilis* cells outlined with different colors. Cells can be tracked by following their respective colors throughout the movie. White arrows indicate cell division events. White and colored dots refer to the new and old poles, respectively. Frame rate is one frame per minute. Original movie data were kindly provided by Prof. Avigdor Eldar and Dr. Jordi van Gestel.
(MP4)

S1 Table. Specific versions used in the environment to run DeLTA 2.0. Package name and respective number of the version that was used for analysis presented in this manuscript, as well as for other working installations.
(DOCX)

Acknowledgments

We thank Virgile Andreani for carefully evaluating the code, Michael Sheets for testing initial versions of DeLTA 2.0, and members of the Dunlop Lab for helpful discussions. Prof. Avigdor Eldar and Dr. Jordi van Gestel generously provided the original images presented in S2 Movie. We thank Dr. Simon van Vliet for useful discussion regarding his datasets.

Author Contributions

Conceptualization: Owen M. O'Connor, Jean-Baptiste Lugagne, Mary J. Dunlop.

Investigation: Owen M. O'Connor, Razan N. Alnahhas, Jean-Baptiste Lugagne.

Methodology: Owen M. O'Connor, Jean-Baptiste Lugagne.

Software: Owen M. O'Connor, Jean-Baptiste Lugagne.

Supervision: Jean-Baptiste Lugagne, Mary J. Dunlop.

Writing – original draft: Owen M. O'Connor, Mary J. Dunlop.

Writing – review & editing: Owen M. O'Connor, Razan N. Alnahhas, Jean-Baptiste Lugagne, Mary J. Dunlop.

References

1. Bakshi S, Leoncini E, Baker C, Cañas-Duarte SJ, Okumus B, Paulsson J. Tracking bacterial lineages in complex and dynamic environments with applications for growth control and persistence. Nature Microbiology. 2021;6. https://doi.org/10.1038/s41564-021-00900-4 PMID: 34017106
2. Fox ZR, Fletcher S, Fraisse A, Aditya C, Sosa S. MicroMator: Open and Flexible Software for Reactive Microscopy. bioRxiv. 2021; 1–9.
3. Rullan M, Benzinger D, Schmidt GW, Milias-Argeitis A, Khammash M. An Optogenetic Platform for Real-Time, Single-Cell Interrogation of Stochastic Transcriptional Regulation. Molecular Cell. 2018; 70: 745–756.e6. https://doi.org/10.1016/j.molcel.2018.04.012 PMID: 29775585
4. Chait R, Ruess J, Bergmiller T, Tkačik G, Guet CC. Shaping bacterial population behavior through computer-interfaced control of individual cells. Nature Communications. 2017;8. https://doi.org/10.1038/s41467-017-00021-9 PMID: 28364116
5. Dal Co A, van Vliet S, Kiviet DJ, Schlegel S, Ackermann M. Short-range interactions govern the dynamics and functions of microbial communities. Nature Ecology and Evolution. 2020; 4: 366–375. https://doi.org/10.1038/s41559-019-1080-2 PMID: 32042125
6. van Vliet S, Dal Co A, Winkler AR, Spriewald S, Stecher B, Ackermann M. Spatially Correlated Gene Expression in Bacterial Groups: The Role of Lineage History, Spatial Gradients, and Cell-Cell Interactions. Cell Systems. 2018; 6: 496–507.e6. https://doi.org/10.1016/j.cels.2018.03.009 PMID: 29655705
7. Bergmiller T, Andersson AMC, Tomasek K, Balleza E, Kiviet DJ, Hauschild R, et al. Biased partitioning of the multidrug efflux pump AcrAB-TolC underlies long-lived phenotypic heterogeneity. Science. 2017; 356: 311–315. https://doi.org/10.1126/science.aaf4762 PMID: 28428424
8. el Meouche I, Dunlop MJ. Heterogeneity in efflux pump expression predisposes antibiotic-resistant cells to mutation. Science. 2018; 362: 686–690. https://doi.org/10.1126/science.aar7981 PMID: 30409883
9. Ronneberger O, Fischer P, Brox T. 2015-U-Net. arXiv. 2015; 1–8.
10. Panigrahi S, Murat D, Gall A le, Martineau E, Goldlust K, Fiche J-B, et al. MiSiC, a general deep learning-based method for the high-throughput cell segmentation of complex bacterial communities. bioRxiv. 2020; 2020.10.07.328666. https://doi.org/10.1101/2020.10.07.328666
11. Lugagne J-B, Lin H, Dunlop MJ. DeLTA: Automated cell segmentation, tracking, and lineage reconstruction using deep learning. Asthagiri AR, editor. PLOS Computational Biology. 2020; 16: e1007673. https://doi.org/10.1371/journal.pcbi.1007673 PMID: 32282792
12. Salem D, Li Y, Xi P, Phenix H, Cuperlovic-Culf M, Kærn M. Yeastnet: Deep-learning-enabled accurate segmentation of budding yeast cells in bright-field microscopy. Applied Sciences (Switzerland). 2021; 11. https://doi.org/10.3390/app11062692
13. Xu YKT, Call CL, Sulam J, Bergles DE. Automated in vivo Tracking of Cortical Oligodendrocytes. Frontiers in Cellular Neuroscience. 2021; 15. https://doi.org/10.3389/fncel.2021.667595 PMID: 33912017

14. Pedone E, de Cesare I, Zamora-Chimal CG, Haener D, Postiglione L, la Regina A, et al. Cheetah: A Computational Toolkit for Cybergenetic Control. ACS Synthetic Biology. 2021. https://doi.org/10.1021/acssynbio.0c00463 PMID: 33904719

15. Rossi NA, el Meouche I, Dunlop MJ. Forecasting cell fate during antibiotic exposure using stochastic gene expression. Communications Biology. 2019; 2: 1–7. https://doi.org/10.1038/s42003-018-0242-0 PMID: 30740537

16. Shao B, Rammohan J, Anderson DA, Alperovich N, Ross D, Voigt CA. Single-cell measurement of plasmid copy number and promoter activity. Nature Communications. 2021; 12. https://doi.org/10.1038/s41467-021-21734-y PMID: 33674569

17. Łapińska U, Glover G, Capilla-Lasheras P, Young AJ, Pagliara S. Bacterial ageing in the absence of external stressors. Philosophical Transactions of the Royal Society B: Biological Sciences. 2019; 374. https://doi.org/10.1098/rstb.2018.0306 PMID: 30967024

18. Stewart EJ, Madden R, Paul G, Taddei F. Aging and Death in an Organism That Reproduces by Morphologically Symmetric Division. Kirkwood T, editor. PLoS Biology. 2005; 3: e45. https://doi.org/10.1371/journal.pbio.0030045 PMID: 15685293

19. Young JW, Locke JCW, Altinok A, Rosenfeld N, Bacarian T, Swain PS, et al. Measuring single-cell gene expression dynamics in bacteria using fluorescence time-lapse microscopy. Nature Protocols. 2012; 7: 80–88. https://doi.org/10.1038/nprot.2011.432 PMID: 22179594

20. Paintdakhi A, Parry B, Campos M, Irnov I, Elf J, Surovtsev I, et al. Oufti: An integrated software package for high-accuracy, high-throughput quantitative microscopy analysis. Molecular Microbiology. 2016; 99: 767–777. https://doi.org/10.1111/mmi.13264 PMID: 26538279

21. Stylianidou S, Brennan C, Nissen SB, Kuwada NJ, Wiggins PA. SuperSegger: robust image segmentation, analysis and lineage tracking of bacterial cells. Molecular Microbiology. 2016; 102: 690–700. https://doi.org/10.1111/mmi.13486 PMID: 27569113

22. van Valen DA, Kudo T, Lane KM, Macklin DN, Quach NT, DeFelice MM, et al. Deep Learning Automates the Quantitative Analysis of Individual Cells in Live-Cell Imaging Experiments. PLoS Computational Biology. 2016. https://doi.org/10.1371/journal.pcbi.1005177 PMID: 27814364

23. Prangemeier T, Reich C, Koeppl H. Attention-Based Transformers for Instance Segmentation of Cells in Microstructures. Proceedings—2020 IEEE International Conference on Bioinformatics and Biomedicine, BIBM 2020. 2020; 700–707. https://doi.org/10.1016/j.copbio.2020.02.002 PMID: 32172160

24. Wang P, Robert L, Pelletier J, Dang WL, Taddei F, Wright A, et al. Robust growth of escherichia coli. Current Biology. 2010; 20: 1099–1103. https://doi.org/10.1016/j.cub.2010.04.045 PMID: 20537537

25. van Gestel J, Bareia T, Tenennbaum B, Dal Co A, Guler P, Aframian N, et al. Short-range quorum sensing controls horizontal gene transfer at micron scale in bacterial communities. Nature Communications. 2021; 12: 1–11. https://doi.org/10.1038/s41467-020-20314-w PMID: 33397941

26. Wen X, Langevin AM, Dunlop MJ. Antibiotic export by efflux pumps affects growth of neighboring bacteria. Scientific Reports. 2018; 8: 1–9. https://doi.org/10.1038/s41598-017-17765-5 PMID: 29311619

27. Ducret A, Quardokus EM, Brun Y v. MicrobeJ, a tool for high throughput bacterial cell detection and quantitative analysis. Nature Microbiology. 2016; 1: 1–7. https://doi.org/10.1038/nmicrobiol.2016.77 PMID: 27572972

28. Linkert M, Rueden CT, Allan C, Burel JM, Moore W, Patterson A, et al. Metadata matters: Access to image data in the real world. Journal of Cell Biology. 2010; 189: 777–782. https://doi.org/10.1083/jcb.201004104 PMID: 20513764

29. McQuin C, Goodman A, Chernyshev V, Kamentsky L, Cimini BA, Karhohs KW, et al. CellProfiler 3.0: Next-generation image processing for biology. PLoS Biology. 2018; 16: 1–17. https://doi.org/10.1371/journal.pbio.2005970 PMID: 29969450

30. Rang CU, Peng AY, Poon AF, Chao L. Ageing in Escherichia coli requires damage by an extrinsic agent. Microbiology (United Kingdom). 2012; 158: 1553–1559. https://doi.org/10.1099/mic.0.057240-0 PMID: 22422756

31. Clark MW, Yie AM, Eder EK, Dennis RG, Basting PJ, Martinez KA, et al. Periplasmic acid stress increases cell division asymmetry (Polar Aging) of Escherichia coli. PLoS ONE. 2015; 10: 1–14. https://doi.org/10.1371/journal.pone.0144650 PMID: 26713733

32. Lee TC, Kashyap RL, Chu CN. Building Skeleton Models via 3-D Medial Surface Axis Thinning Algorithms. CVGIP: Graphical Models and Image Processing. 1994; 56: 462–478. https://doi.org/10.1006/cgip.1994.1042

33. Sommer C, Straehle C , Ullrich K, Hamprecht F a. ILASTIK: INTERACTIVE LEARNING AND SEGMENTATION TOOLKIT Heidelberg Collaboratory for Image Processing (HCI), University of Heidelberg. Eighth IEEE International Symposium on Biomedical Imaging (ISBI). 2011; 230–233.

34. van Vliet S, Dal Co A, Winkler AR, Spriewald S, Stecher B, Ackermann M. Spatially Correlated Gene Expression in Bacterial Groups: The Role of Lineage History, Spatial Gradients, and Cell-Cell Interactions. Cell Systems. 2018; 6: 496–507.e6. https://doi.org/10.1016/j.cels.2018.03.009 PMID: 29655705

35. Lee TS, Krupa RA, Zhang F, Hajimorad M, Holtz WJ, Prasad N, et al. BglBrick vectors and datasheets: A synthetic biology platform for gene expression. Journal of Biological Engineering. 2011; 5: 15–17. https://doi.org/10.1186/1754-1611-5-15 PMID: 22059903