LivecellX: Corrective Deep Learning for Object-Oriented Single-Cell Analysis in Live-Cell Imaging

Ke Ni[1,2], Gaohan Yu[3], Zhiqian Zheng[3], Yong Lu[1], Sophia Hu[1,2], Dante Poe[1,2], Sijie Zhang[1], Mia Sanborn[4], Mostofa Uddin[2,4], Zilin Wang[1], Shiman Zhou[1], Yanshuo Chen[5], Xueying Zhan[4], Weikang Wang[6,7*], Jianhua Xing[1,3,8*]

[1]Department of Computational and Systems Biology, University of Pittsburgh, Pittsburgh, 15232, PA, USA.
[2]Joint CMU-Pitt Ph.D. Program in Computational Biology, University of Pittsburgh, Pittsburgh, 15232, PA, USA.
[3]Department of Physics and Astronomy, University of Pittsburgh, Pittsburgh, 15232, PA, USA.
[4]Ray and Stephanie Lane Computational Biology Department, Carnegie Mellon University, Pittsburgh, 15213, PA, USA.
[5]Department of Computer Science, University of Maryland, College Park, 20742, MD, USA.
[6]Institute of Theoretical Physics, Chinese Academy of Sciences, Beijing, 100190, China.
[7]School of Physical Sciences, University of Chinese Academy of Sciences, Beijing, 100049, China.
[8]UPMC-Hillman Cancer Center, University of Pittsburgh, Pittsburgh, 15232, PA, USA.

*Corresponding author(s). E-mail(s): wangwk@itp.ac.cn; xing1@pitt.edu;

Abstract

Live-cell imaging uniquely captures single-cell dynamics in space and time, but robust analysis is limited by segmentation and tracking errors that accumulate across frames. We present LivecellX, a deep-learning–based pipeline that integrates instance-level segmentation error correction with trajectory refinement, leveraging temporal context to recover accurate cell tracks. LivecellX also introduces a benchmark dataset with detailed annotations of common error classes, providing a resource for method development and evaluation. Beyond error correction, the framework incorporates modules for classifying biological processes, reconstructing cell lineages, and analyzing dynamic behaviors. Users can interact with the system programmatically or through a Napari-based graphical interface, enabling flexible integration into diverse workflows. By coupling error-aware correction with comprehensive lineage and dynamics analysis, LivecellX establishes an open, extensible platform that advances the accuracy and scalability of live-cell imaging studies.

Keywords: live-cell imaging, corrective segmentation network, lineage reconstruction, single-cell trajectory

1 Introduction

Advances in single-cell techniques have transformed the characterization of cellular states-phenotype, transcriptomic, proteomic, epigenomic, and metabolic-and their dynamics. These methods are broadly divided into sequencing- and imaging-based, or alternatively into destructive snapshot approaches and nondestructive methods that enable longitudinal cell tracking. Most sequencing- and multiplex staining-based imaging techniques are destructive [1–8].

Unlike computational inference of cell-state transitions from snapshots, live-cell imaging directly captures spatial and temporal dynamics [9–12], but its adoption lags behind destructive high-throughput methods due to technical challenges. One main challenge is cell segmentation, which must be highly accurate to enable reliable tracking and quantification of features such as morphology and protein distribution [13]. Deep learning, particularly convolutional neural networks and vision transformers, has enabled large-scale segmentation [8, 14–19], and tools like CellPose and μSAM represent current state-of-the-art. However, no method achieves perfect accuracy [20]; segmentation errors, especially under-segmentation(under-seg) and over-segmentation(over-seg), remain common and propagate into tracking, where performance depends heavily on segmentation quality (Fig.1a)[21–25].

Consider a live-cell imaging experiment that images 10^2–10^4 cells simultaneously every 10 min for a duration of 16 hours, generating movies with ~ 100 frames. With a segmentation accuracy of 98%, the probability of tracking a trajectory without any segmentation errors is merely approximately $0.98^{100} \approx 0.13$(Fig. 1b). For a seven day experiment, the accuracy further reduces to $0.98^{1008} \approx 1 \times 10^{-9}$. With a large long-time imaging dataset, e.g., the above-mentioned seven-day experiment with a total of $\sim 10^5$–10^7 cells, it is impractical to identify and correct a fraction of $\sim 10^3$–10^5 mis-segmented cells through visual inspection. In applications, label-free live-cell imaging emerges as an appealing alternative to fluorescent imaging because of the reduced photo-toxicity and no requirement of genetic engineering [11]. Due to the lack of corresponding training data, segmentation accuracy is typically even lower than that achieved with fluorescent images.

Even worse is that segmentation errors often intertwine with single cell trajectory tracking. For instance, one segmentation error in time frame t together with movement of cells may generate errors not only in that specific cell trajectory but also in tracking errors in trajectories of neighboring cells (Fig. 1c).

Cell division also increases the complexity of extracting trajectories [26]. Identification of cell division is the key step in lineage tracing that is crucial for analyzing long-term live-cell imaging data [27]. But the scarcity of cell division poses challenges for deep learning models that rely on large and balanced training datasets. For cell dynamics analyses one needs to extract high-dimensional features and choose appropriate representations for cell states [10, 28, 29]. However, limitation in techniques of microscopy and labeling imposes challenges on the widely used multiplex fluorescent labeling for feature extraction in snap-shot imaging experiment. Furthermore, fluorescence labeling can induce significant photo-toxicity in cells during long-term live-cell imaging [27].In recent years, label-free transmitted light imaging emerges as an alternative imaging modality, with significantly enhanced reliability and repeatability of experiments, particularly for long-term live-cell imaging studies.

For label-free or low-plex fluorescent imaging, morphological and textural features are widely used to represent cell states and analyze trajectory dynamics [7, 10, 28, 30–33]. Hence, software package should include modules for label-free feature extraction and dynamical systems theory-based trajectory analysis[10, 34, 35].

While many computational tools exist for snapshot imaging [7, 8], specialized packages for live-cell imaging remain scarce. Although several solutions support single-cell tracking and trajectory analysis that tackle some but not all the associated challenges simultaneously [26, 36–39]. For instance, most methods rely on stained nuclei [40] and manually curated features, with only a few applying deep learning [26, 41]. The growing scale of live-cell imaging further demands computational efficiency, memory

optimization, and dedicated modules for dynamical systems theory-based single cell trajectory analysis.

In this study, we present LivecellX, a comprehensive workflow for analyzing single-cell trajectories from time-lapse imaging datasets. The contribution of LivecellX is threefold. First, it provides an automated procedure to detect and correct under-seg and over-seg errors as well as tracking errors by developing a deep-learning-based method, Corrective Segmentation Network (CS-Net), and integrating temporal information. Second, the LivecellX architecture is designed to transform the analysis task from an image-matrix-oriented processing paradigm to a single-cell object-oriented paradigm. Based on the LivecellX architecture, we integrated methods and designed programming interfaces for tracking, extracting features, detecting biological process, lineage reconstruction, and dynamics analyses. Finally, LivecellX is integrated with Napari to allow interactive user-in-the-loop analyses of live-cell data trajectory objects and cell objects.

2 Results

2.1 Overview of the LivecellX architecture and workflow

LivecellX is a comprehensive framework for live-cell imaging data analyses (Fig. 1d). It is designed to improve cell segmentation and tracking accuracy based on the additional temporal information between consecutive frames in live-cell imaging, and simplify trajectory dynamics extraction and complex cellular process analysis.

LivecellX takes cell segmentation and trajectory tracking results from other methods as input (Fig. 1e), saves them as a collection of single cell trajectories (Fig. 1f). The segmentation and trajectory results are fed into and iterate between an inconsistency identification module and the CS-Net (Fig. 1g). The identification module exploits the temporal information between consecutive frames for identifying candidates of potentially under- or over-seg cells, and feeds the information to CS-Net. CS-Net is a context-aware, multi-scale machine learning architecture designed to automatically reduce segmentation inaccuracies. It has been pretrained specifically for identifying under-seg and over-seg. With the input from the identification module, CS-Net examines the candidate cells and their neighborhoods for segmentation refinement, then returns to the identification module for reexamination.

After the identification module and CS-Net reach a self-consistent result of high quality trajectories, LivecellX provides three downstream analysis modules. A biological process detection module detects biological processes under interest such as mitosis and apoptosis. Correct detection of mitosis enables reconstruction of cell lineages and downstream analyses (Fig. 1h left). A cell feature extraction module optimized for efficiency via parallelization quantifies high-dimensional features like morphology and texture features and stores the information in single-cell objects (Fig. 1h middle). With a third dynamics analysis module, one represents single cell states and trajectories in a high-dimensional feature space (Fig. 1h right). The quantitative representation enables users to uncover underlying patterns in cell behavior, identify dynamic cellular states, and cluster cells based on temporal representation to reveal processes such as cell cycle progression, response to external stimuli or drug treatments, and disease-related phenotypic changes, etc. All these downstream analyses are implemented based on an object-oriented data framework (Fig. 1i). The framework significantly increases the efficiency of information extraction and manipulation on trajectories.

To ensure robustness and usability, we developed an asynchronous graphical user interface (GUI) based on Napari [42], allowing human-in-the-loop surveillance and intervention at any stage of the analyses. The GUI includes a visualization window that displays microscopy images alongside corresponding segmentation masks and annotations (Fig. 1j). When users encounter errors in specific trajectories—such as segmentation drift or mitotic mislabeling—the GUI enables both visual verification and manual correction (Fig. 1j left and middle, Extended Data Fig. 1a), as well as trajectory-level operations like deletion or refinement (Fig. 1j right,Extended Data

Fig. 1b). The LivecellX data structures enable the development of extensible, multi-purpose GUIs. For example, two Napari windows can be synchronized and controlled by LivecellX to streamline annotation: when the user clicks 'annotate,' both windows automatically advance to the next time frame and highlight the corresponding pair of cell masks. Annotation labels are also easily configurable at launch.

2.2 Correction segmentation errors with the CS-Net

Despite recent development of deep-learning-based segmentation methods, segmentation errors are generally unavoidable when processing large imaging datasets. Typical segmentation errors include under-seg where two or more cells are incorrectly segmented as one mask (Fig. 2a top), and over-seg where one cell is incorrectly segmented into two or more masks (Fig. 2a middle) [43–45]. In addition, we defined a subtype of over-seg error termed over-seg dropout error for cases where only a portion of the single-cell is segmented while other portions are not (Fig. 2a bottom).

The candidate segmentation errors are identified using temporal context (see next section) and cropped as region of interest (ROI) from the original images (Fig. 2b). The trained CS-Net takes each ROI– consisting of a candidate cell, its neighborhood, and the segmentation mask –as input and outputs a corrected mask as well as assigns the segmentation status to one of four categories: under-seg, over-seg, over-seg dropout, and no error (negative) (Fig. 2b). An ROI is labeled as negative if the cell is correctly segmented and requires no correction. The CS-Net model comprises a feature encoder—based on either a vision transformer or a convolutional neural network which compresses the ROI into a latent representation, followed by two subnetworks: a segmentor and a classifier (Fig. 2c).

2.2.1 Training of CS-Net

CS-Net needs to be pretrained on curated training and testing datasets independent of inputs from the identification module. A training dataset includes ROI samples of all the four categories mentioned in the above section. To fully leverage annotated data, we implemented synthetic data generation algorithms to introduce both under-seg and over-seg samples, particularly to address class imbalance in our datasets (Extended Data Fig. 2a-e). Having negative samples in the training dataset enhances the CS-Net robustness from altering accurate segmentation results misidentified by the identification module.

A key challenge for CS-Net is to target only erroneous cells within an ROI (Fig. 2d). CS-Net differs from typical segmentation models by concentrating on potentially incorrect cell masks through adding an additional image channel, i.e., a *focus mechanism*. By focusing on individual cell cases, CS-Net allows for precise training and development of specific performance metrics, enhancing control over cell subpopulations. While the focus channel can be the mask or label mask, we found that the Euclidean distance transform (EDT) of the mask (Fig. 2e)[46] work the most robustly on guiding the deep learning model to concentrate during correction (Section 5.2.2). Furthermore, we implemented a normalization technique that scales the EDT values to a fixed range of 1 to 5, producing a contour-map-style representation. This normalized input helps CS-Net prioritize pixels that require correction (Fig. 2e right), and mitigate numerical instability of varying EDT pixel values caused by different cell sizes. The normalized EDT mask and the raw label-free channel image are then jointly fed to CS-Net for correction. During postprocessing, the normalized EDT values further facilitate selecting an appropriate threshold for generating binary masks (Section 5.2.2).

For under-seg, the focus channel highlights regions of individual cells that need separation, whereas over-seg requires identifying multiple fragments of a single cell for merging, which typically relies on manual annotation or auxiliary detection algorithms. To avoid these complexities, we augmented the training data with over-seg dropout samples, each containing a single fragment from an over-seg case (see Section 5.2), enabling CS-Net to reconstruct full cell masks from partial, fragmented

inputs. Combined with the focus mechanism, CS-Net also employs dataset processing strategies—such as varying ROI padding to include different neighborhood contexts—that enhance generalization and robustness across diverse input sizes and spatial resolutions(Extended Data Fig. 2f).

2.2.2 Evaluating CS-Net

Fig. 2f shows representative corrections of under-seg, over-seg, and over-seg dropout. These inputs span varying scales and background padding, all of which CS-Net robustly handled. The model generalizes across different types of microscopy images with high confluence of cells (Extended Data Fig. 3a). Additionally, CS-Net resolves non-trivial errors beyond canonical under- and over-seg like adjacent-cell merges, boundary ambiguities, three-cell under-seg, and uncategorized mis-segmentation (Extended Data Fig. 3b).

We evaluated segmentation performance using four metrics: cell match rate, correction success rate, intersection over union (IoU) calculated on cell contours (morphology), and root mean square error (RMSE) of EDT masks (Extended Data Fig. 3c-e; see Methods for definitions). Our evaluation employs instance-level mask-based metrics that are more sensitive to mask errors.

Statistical analysis of mask-based metrics showed that CS-Net with the focus mechanism significantly improved segmentation quality (Fig. 2g). CS-Net successfully corrected nearly all over-seg error cases. In under-seg scenarios, CS-Net achieved strong performance across three of the four evaluation metrics, with the exception of the conventional binary mask IoU, which is commonly used in the field (Fig. 2g). This metric is not well suited for evaluating under-seg, as erroneous and ground-truth masks often appear nearly identical at the pixel level despite representing different cell instances(Fig. 2f). Consequently, even a perfect correction by CS-Net may result in minimal or no change in the binary mask IoU. In contrast, RMSE of EDT masks, cell match rate, and correction success rate provide more sensitive and informative assessments for under-seg cases by directly evaluating instance correspondence and morphological accuracy.

We compared the performance of CS-Net with and without the focus mechanism under different conditions (Fig. 2h–j). For under-seg and negative cases, the performance gap between the two models increases as padding size grows, reflecting the presence of more cells in the label-free image crop. For over-seg cases, the gap widens as the IoU threshold becomes more stringent, demonstrating the advantage of the focus mechanism under stricter criteria. When no padding is applied, however, over-seg crops often contain only a single cell, in which case the focus mechanism provides little benefit. Overall, CS-Net with the focus mechanism achieved consistently higher cell match percentages across all conditions, maintaining robust performance across varying background sizes, as controlled by different padding values from raw imaging data (Extend Data Fig. 3 f-h). Notably, the performance gap favoring the focus mechanism expanded with larger padding sizes, underscoring its effectiveness in handling diverse spatial contexts. The comparison of correction success rates at different padding size, IoU thresholds, microscopy modalities and error types on in different data subsets also reveals the robustness of this method(Extend Data Fig. 3 i-j).

The accuracy of the classifier was also higher when CS-Net is equipped with the focus mechanism. This aligns with intuition: without the focus mechanism, CS-Net lacks information about which specific cell within the ROI should be classified and segmented, leading to reduced performances (Fig. 2k).

Across encoder backbones, we observed no consistent advantage of ViT-based over convolutional neural network (CNN)-based backbones when parameter counts were comparable. Instead, scale dominated: larger encoders yielded better corrections. Within DINOv3 backbones, a fine-tuned large variant (ViT-7B/16) substantially outperformed fine-tuned small variants (e.g., ViT-S/16): correction success rate with small ViT models could drop from ~ 0.7 to ~ 0.3 on our benchmarks, whereas the larger model maintained high success. A similar scale effect held for CNN backbones, with

larger models consistently outperforming smaller ones under matched training settings (see Methods for configurations).

2.3 Linkage correction in single cell trajectories with CS-Net

Cells are deformable, often exist in high-density environments, and can divide, disappear, or occlude each other. While existing tracking algorithms have been adapted to account for these biological constraints, they generally assume that the input segmentation masks are accurate [47–49].

However, segmentation errors cause several typical linkage distortions between single-cell masks including cascade of wrong linkages involving neighboring cells caused by under-seg of two cells from different trajectories(Fig. 3a), decreased trajectory quality and erroneous linkages caused by unassigned over-seg fragments (Fig. 3b), missing frames due to absent cells (Fig. 3c, Extended Data Fig. 4a, details in Methods). Linkage errors can be further complicated when more neighboring cells are involved.

Robust lineage reconstruction requires jointly correcting segmentation masks and tracking linkages, motivating integrated approaches that iteratively refine both. In CS-Net, candidate errors are first detected and cropped as ROIs. Note that there are typically strong overlap between correct masks across consecutive frames, and segmentation errors appear as abrupt changes in overlap or morphology (Fig. 3a-c). To capture these, an inconsistency detection module (TD-1) filters temporal inconsistencies using IoU and Intersection over Minimum (IoMin) metrics (Extended Data Fig. 4b). Each under-seg candidate was represented as a triplet $\{\text{cell}_t^u, \text{cell}_{t+\Delta t}^1, \text{cell}_{t+\Delta t}^2\}$, where cell_t^u denotes the under-seg mask. Each over-seg candidate was represented as a pair $\{\text{cell}_t^o, \text{cell}_{t+\Delta t}\}$, where cell_t^o denotes the over-seg mask (Extended Data Fig. 4c). Candidates meeting the error criteria are cropped with surrounding regions and passed to CS-Net for correction (Fig. 3d). The candidate triplets are affected by the threshold values of IoU and IoMin scores (Extended Data Fig. 4d-e). In addition, the efficiency of detection is affected by situations like mitosis and cells with high motility.

Testing on multiple datasets revealed that the TD-1 method alone was insufficient for detecting segmentation errors, particularly when errors occurred in consecutive frames along a single-cell trajectory (Extended Data Fig. 4f). To address this limitation, we developed a multi-iteration, trajectory-level algorithm that employs CS-Net to iteratively correct both segmentation and track linkage errors (Fig. 3e). In each iteration, trajectory-level errors are identified, corrected by CS-Net, and the corresponding trajectories updated. This process ensures that the first error in a sequence of persistent errors is resolved in the first iteration, while subsequent errors are corrected in later iterations. Typical linkage errors—caused by under-seg (Fig. 3f, left), over-seg (Fig. 3f, middle), and absent cells (Fig. 3f, right)—are successfully corrected by this approach.

Validation on A549 trajectories showed that, after fifty iterations, the number of attempted and resolved under-seg corrections reached saturation (Fig. 3g). As expected, a clear gap remained between potential errors flagged by TD-1 and true errors corrected by CS-Net, reflecting that CS-Net can detect false positives included in TD-1 detection (Extended Data Fig. 4g-h). The ROC analysis (Fig. 3h) further demonstrated that CS-Net with trajectory inconsistency detection achieved an AUC of 97%, outperforming the 87% AUC from TD-1 features alone. Visual inspection confirmed that all true over-seg errors were correctly identified and corrected.

Next, we compared trajectory lengths before and after our correction procedure (Fig. 3i). For each corrected trajectory, we calculated the relative increase in length. Across all tested settings of the maximum age parameter(*max gap*)in the tracking algorithm (Methods), the median relative increase exceeded 25%, demonstrating the robustness of our correction method (Fig. 3i). Notably, when max gap was small, the relative increase approached 40%, as shorter *max gap* values tend to produce more fragmented trajectories. In line with this, the total number of trajectories decreased substantially after correction, reflecting the successful linking of short trajectory fragments (Fig. 3j).

To further assess performance, we quantified the trajectory vacancy rate, defined as the proportion of missing frames due to factors such as segmentation errors or cells moving out of the field of view. Approximately half of these vacancies were resolved following correction, regardless of the *max gap* setting, underscoring the method's effectiveness in improving trajectory continuity and quality (Fig. 3k).

2.4 A biological process detection module allows user-specified cell biology event-aware image analyses

Tracking cells in live-cell images is uniquely challenging due to biological processes such as division and apoptosis. Cell division, in particular, adds complexity to single-cell dynamics and lineage analysis. When a mother cell divides, its trajectory should terminate while two daughter trajectories are initiated, with the mother–daughter relationship preserved for lineage reconstruction. Without accurate mitosis detection, tracking algorithms may incorrectly link trajectories or miss lineage relationships (Fig. 4a).

Therefore, we developed a module to identify frames where user-specified biological processes occur. As an example, we trained a network to detect mitosis, which is marked by rapid morphological changes; in 2D cultures, both mother and daughter cells appear small and round (Fig. 4b).

We integrated video-based deep learning methods by leveraging full temporal and visual context. Two frameworks — CNN- and transformer-based (Fig. 4c) — were fine-tuned from MMAction video-classification models [50] and adapted to treat mitosis detection as a single-cell trajectory classification task. Detected events are stored in the LivecellX trajectory structure, though manual annotation may still be required for data from different microscopes or conditions. Alternatively, users can identify divisions via the GUI.

Fig. 4d shows an A549 single-cell lineage spanning three generations, revealing differences in motility and cell cycle duration among sister and cousin cells. LivecellX also enables statistical analysis of cell cycle durations (Fig. 4e) and visualization of multiple reconstructed lineages across generations, including spatial positions (Fig. 4f, left) and displacements relative to the initial frame (Fig. 4f, right).

2.5 Feature extraction and dynamics analysis modules provide tools for cell dynamics studies

A key downstream task in LivecellX is extracting multiplex features from fluorescent and/or label-free images to represent single-cell trajectories in multidimensional feature space. These features enable visualization, interpretation, and modeling of cellular dynamics. LivecellX integrates tools such as `scikit-image` and `mahotas` to extract morphological and texture-based descriptors (e.g., Haralick features), which strongly correlate with cell states and capture dynamic changes over time. Importantly, many morphological features can be quantified from transmitted-light images, avoiding the limitations of fluorescence labeling, including restricted marker capacity and phototoxicity.

As a demonstration, we extracted multidimensional morphological and texture features from time-lapse images of two cell lines, MCF10A and A549. Feature distributions differed between the lines (Fig. 5a), and cell line–specific clusters emerged in t-SNE and UMAP embeddings (Fig. 5b). These embeddings correlated with specific morphological and texture features(Fig. 5c, Extend Data Fig. 5a), whose contributions were further examined by principal component analysis (PCA) (Extended Data Fig. 5b).

LivecellX enhances the quality of extracted trajectories in long-term live cell imaging and enables exploration of dynamic changes along each trajectory. The original images at different time points of a typical single cell trajectory can be retrieved conveniently with data framework of LivecellX (Fig. 5d left). The morphology features including cell area, major axis length(MAL) and eccentricity are shown as time series

and can be analyzed in further study(Fig. 5d right). The framework also supports intuitive selection and inspection of individual cells across time points.

In addition to morphology and texture features, LivecellX integrates an active shape (AS) model to represent cell contours [32](Fig. 5e). For example, a representative MCF10A cell trajectory is shown in pixel space with time-colored morphology colored (Fig. 5e left). After alignment the contours are represented with the AS model which extracts multi-dimensional features from single cell contours(Fig. 5e middle), and PCA yields orthogonal modes that describe and reconstruct cell shapes (Fig. 5e right, Extend Data Fig. 5c). Single cell trajectories can then be represented in PCA coordinates (Fig. 5f), capturing dynamic changes in morphology [10, 32]. Applying this model across all trajectories reveals distinct shape dynamics between A549 and MCF10A cells (Fig. 5g).

To exploit the power of machine learning for capturing richer image features beyond handcrafted feature-based methods, we integrated a machine-learning–based feature extractor, Harmony, a variational autoencoder (VAE) designed to disentangle semantic content from transformations and yield invariant, interpretable embeddings [51] (Fig. 5h).

Each cell was represented by a 512-dimensional Harmony embedding, which, when projected using UMAP or t-SNE, yielded clusters comparable to those obtained from traditional morphological and texture features(Fig. 5i, Extended Data Fig. 5d). However, Harmony embeddings contained more uncorrelated features, likely due to their higher information content and the non-linear encoder structure (Extended Data Fig. 5e). To interpret these latent representations, we correlated Harmony embeddings with morphological descriptors, finding strong associations—particularly between the standard deviation component of Harmony embeddings and several morphological and Haralick texture features (Fig. 5j, Extended Data Fig. 5f)

Cellular dynamics can be studied through large collections of single-cell trajectories by reconstructing the underlying dynamic manifold via time-delay embedding, as guaranteed by Takens' theorem [52]. For a cell at time t, this embedding concatenates feature vectors over the range $(t - \Delta t,\ t + \Delta t)$ and can be built from morphology, texture, ASM, or VAE embeddings (Fig. 5k). Using Harmony features, time-delay embedding of a cell cycle trajectory revealed the expected cyclic structure (Fig. 5l).

2.6 LivecellX is a portable single-cell object-oriented framework with Napari interface

Current image-centric live-cell imaging workflows are computationally inefficient, particularly with heterogeneous formats and large datasets, as they rely on matrix-based operations that complicate object tracking. LivecellX overcomes these challenges with an object-oriented, cell-centric design that organizes data around single-cell objects, assembled post-segmentation and integrated into trajectories for efficient analysis (Fig. 6). The framework operates across three levels: (1) Single-cell objects, encapsulating features, metadata, and subcellular details with interfaces compatible with tools like Cellpose and StarDist [8, 53, 54]; (2) Single-cell trajectories, linking objects over time with metadata such as velocity and lineage for automated reconstruction; and (3) Trajectory collections, which organize all trajectories for large-scale statistical analyses and cross-dataset integration. The emphasis on cell-centric data processing reduces repetitive computations and enhances data interpretability across diverse experimental conditions, enabling large-scale, multi-dataset integration.

A Napari-based GUI leverages the above structure to provide interactive, programmatic access for inspecting, editing, and correcting results, particularly at the trajectory level (Extended Data Fig. 6a–b). LivecellX further optimizes data management through dynamic structures and caching mechanisms (e.g., LRU cache), reducing redundant computation and disk access (Extended Data Fig. 6c).

We integrated LivecellX with PyTorch and different machine learning method to enable on-the-fly data generation and stochastic augmentation from single-cell or trajectory objects (Extended Data Fig. 6c). For instance, a diffusion model is trained

for generating single cell images and related masks for training as well as other analyses(Extended Data Fig. 6d).

3 Discussion

In this study, we present LivecellX, a comprehensive pipeline for analyzing single-cell dynamics from time-lapse imaging data. Built on a single-cell object-oriented framework, LivecellX integrates segmentation, tracking, feature extraction, and trajectory analysis while minimizing technical overhead and coding complexity. To address common challenges such as inaccurate segmentation, tracking errors, and lineage reconstruction, it combines automated correction with two fallback options: invalidating those single cell objects or trajectories containing errors, or manual intervention with GUI to manually correct segmentation errors or other issues that algorithms were unable to resolve. LivecellX provides a new method for resolving the segmentation and tracking errors at the same time, while the segmentation and tracking outputs can be generated with classical methods. One alternative way is performing segmentation and tracking simultaneously [25].

Leveraging machine learning, LivecellX reduces labeling burden by pretraining segmentation on public cell image datasets and employing semi-automatic algorithms for labeling incorrect segmentation and key events such as mitosis and apoptosis. LivecellX integrates transmitted-light and fluorescence features and leverages deep learning advances to extract additional information from label-free images such as organelle identification [55, 56]. In future iterations of LivecellX, we aim to incorporate more features to further enhance its functionality. Therefore, LivecellX is designed as an open, extensible framework. It supports future advances in segmentation, tracking, and integration with other analysis pipelines, enabling more robust and comprehensive studies of cellular dynamics.

4 Figures

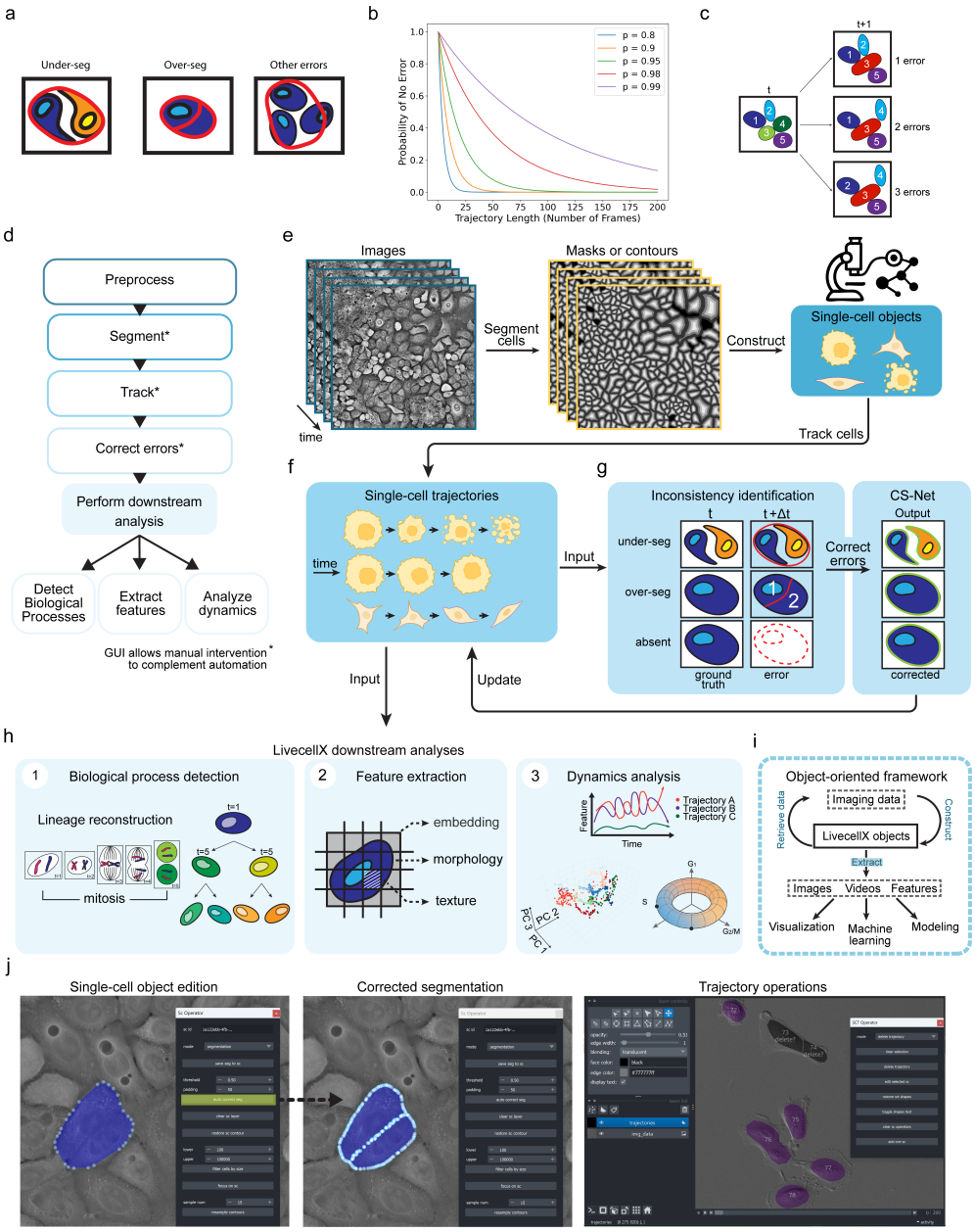

Fig. 1 Overview of the LivecellX framework for analyzing single-cell dynamics from live-cell imaging. **a**, Typical segmentation errors in live-cell imaging, including under-segmentation, over-segmentation, and other errors. Ground-truth cell boundaries (black) and predicted contours (red) are shown schematically. **b**, Probability curves illustrating how per-frame segmentation error rates affect the likelihood of obtaining an entirely error-free trajectory as a function of trajectory length. Lines correspond to different per-frame accuracies (80, 90, 95, 98, and 99%), highlighting the rapid decay of error-free probability in long-term imaging even at high accuracy. **c**, Schematic showing how a single segmentation error can propagate into linkage errors in neighboring trajectories. At frame t, cells 3 and 4 are distinct, but at frame t+1 they are merged due to under-seg. Colors indicate ground-truth linkages, while numbers show tracking algorithm predictions. **d**, The LivecellX pipeline integrates segmentation, tracking, error correction, biological process detection, and trajectory-level analysis. **e**, Starting from raw images, cells are segmented into single-cell objects. **f**, Single-cell objects are tracked to form trajectories. **g**, Segmentation errors (under-seg, over-seg, or absent cells) are detected by inconsistencies between consecutive masks and corrected by CS-Net, which takes both the raw image and erroneous mask as input. **h**, LivecellX supports downstream analyses, including lineage reconstruction (left), feature extraction (middle), and dynamics analysis (right). **i**, An object-oriented data framework enables flexible operations in LivecellX. **j**, The GUI supports multiple levels of user interaction, including single-cell object editing (left), segmentation correction (middle), and trajectory operations (right).

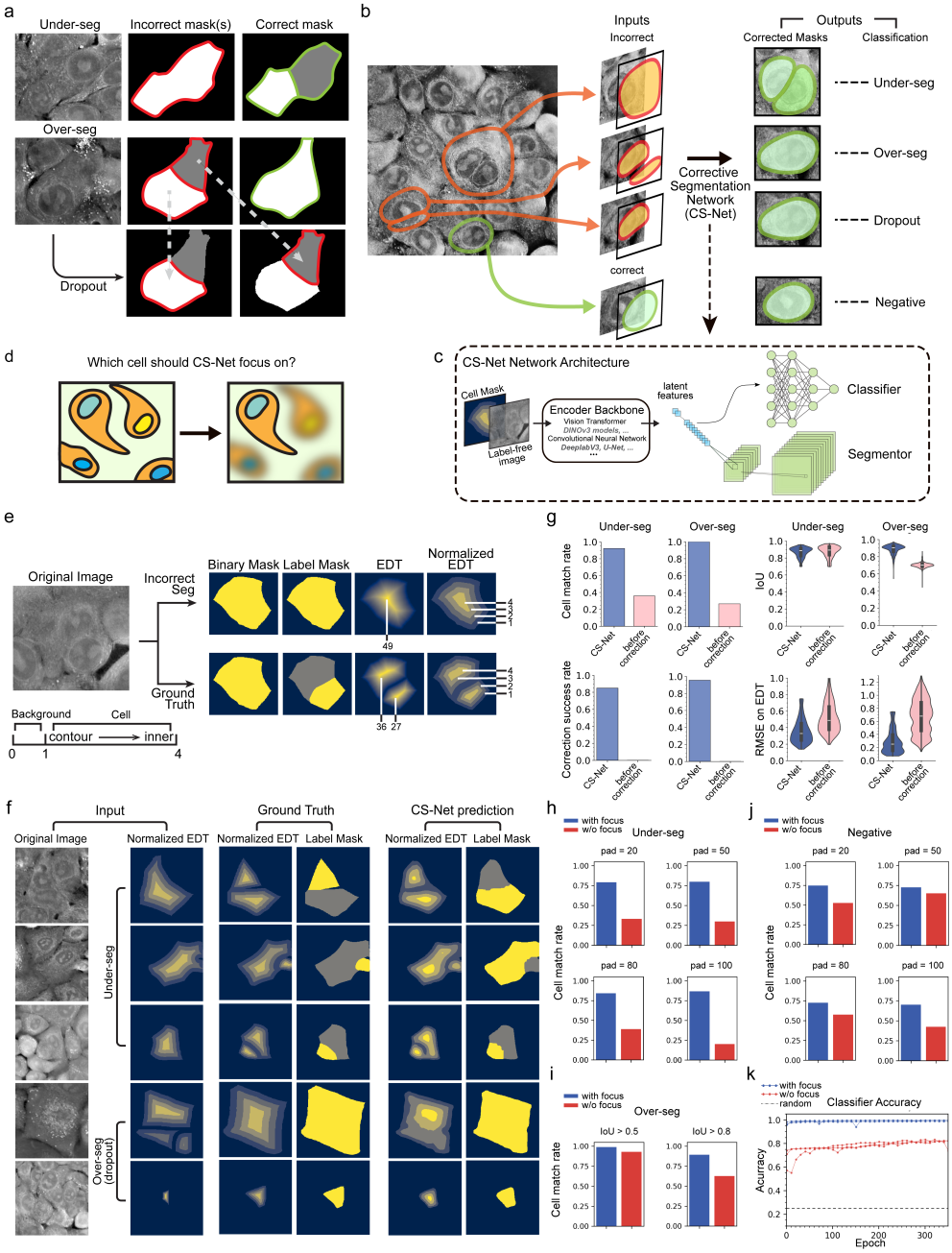

Fig. 2 Corrective segmentation and error classification with CS-Net in LivecellX. a, Representative examples of segmentation errors in live-cell imaging, including under-seg (top), over-seg (middle), and over-seg dropout cases (bottom). For each, the original image, incorrect mask(s) (red), and correct mask(s) (green) are shown. **b**, Schematic of the CS-Net correction process. Candidate cell masks (incorrect and correct) are extracted from label-free images and input into the CS-Net, which outputs a corrected mask and classifies the error as under-seg, over-seg, dropout, or negative (correct). **c**, CS-Net architecture comprising dual-branch encoders for cell mask and label-free image, followed by segmentation decoder and input-type classifier. **d**, Illustration of the focus mechanism: among adjacent or overlapping cells, the focus mechanism enables CS-Net to correct the designated cell instance. **e**, Different mask types used as the focus mechanism including binary mask, label mask, EDT, and normalized EDT for both ground truth and prediction. **f**, Qualitative examples of CS-Net correction for under-seg, over-seg and over-seg dropout cases. Each row shows the original image, input normalized EDT, ground-truth mask, and CS-Net prediction. **g**, Quantitative evaluation of CS-Net on under-seg and over-seg errors. Metrics shown include correction success rate, cell match rate, IoU, and RMSE of EDT. **h–j**, Cell match rate analysis for under-seg(**h**), over-seg(**i**), and negative (correct)(**j**) cases across varying padding sizes, illustrating consistent performance gains with the focus mechanism; comparison of cell match rates for over-seg at different IoU thresholds (**i**). **k**, Training curves for CS-Net classifier accuracy, demonstrating robust convergence above the random baseline. Classification performance is consistently improved by the focus mechanism.

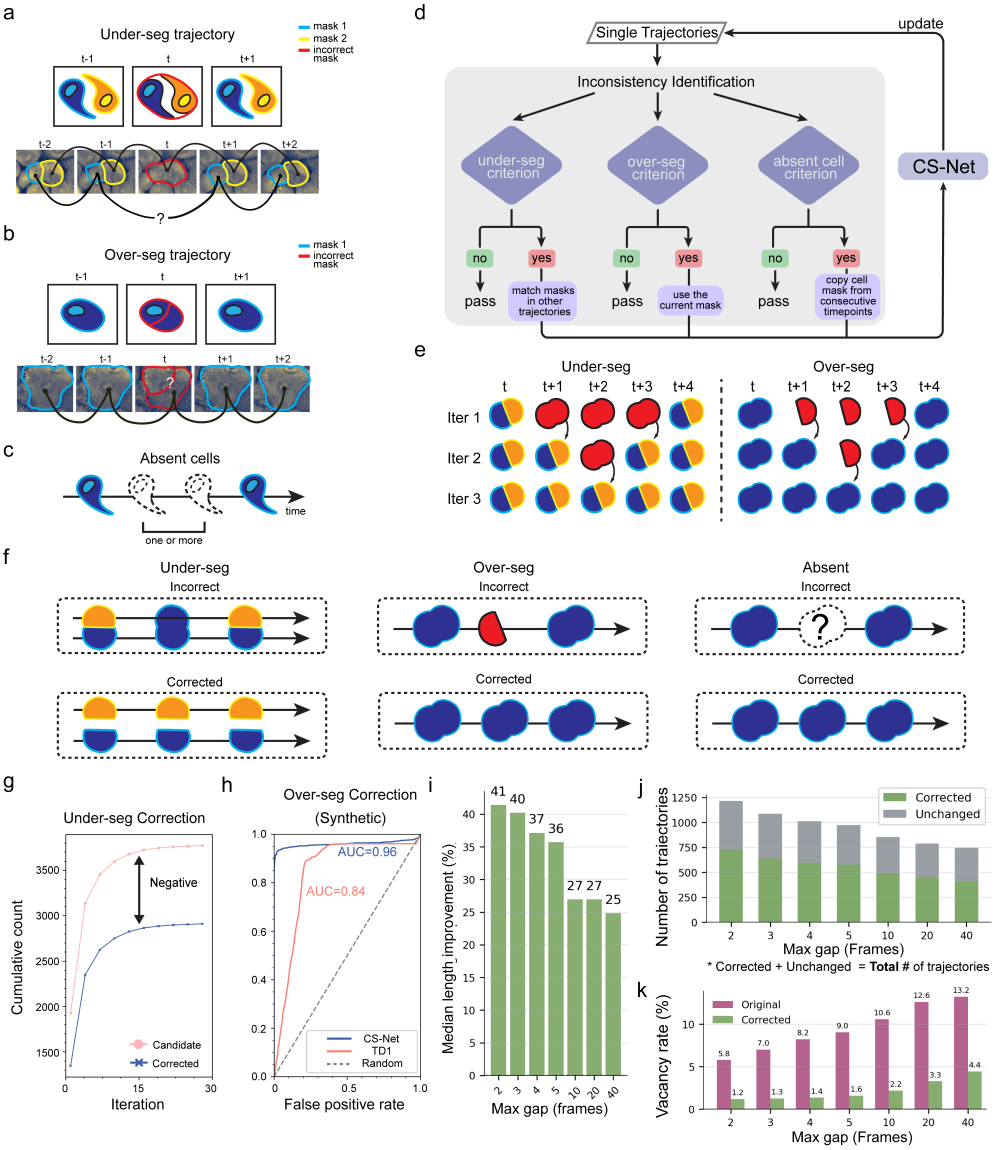

Fig. 3 Tracking correction using CS-Net in LivecellX. **a**, Typical tracking error caused by under-seg. Under-seg results in two trajectories merging into one mask at a given time point, causing a missing frame in one trajectory and an incorrect cell mask at this frame in the other. **b**, Typical tracking error caused by over-seg. A trajectory is split at a single time point, leading to sharp morphological changes and short-lived tracklets due fragmented masks. **c**, Absent cells result in missing frames in the trajectory. **d**, Flowchart of the trajectory correction pipeline. Each trajectory is screened for temporal inconsistencies. Candidate frames are classified as under-seg, over-seg, or absent cell using dedicated criteria, and then corrected by CS-Net. Corrected masks are iteratively updated within the trajectory. **e**, Illustration of multi-iteration correction for persistent under-seg (left) and over-seg (right) errors across several consecutive frames. **f**, Examples of linkage updates in single-cell trajectories before (incorrect) and after (corrected) segmentation error correction for under-seg (left), over-seg (middle), and absent cell errors (right). Arrow represents time flow. **g**, Cumulative counts of under-seg candidates and successfully corrected cases over multiple correction rounds. The difference reflects negative candidates identified by temporal inconsistency screening. Note that TD-1 is designed to have potential negative candidates for subsequent CS-Net identification. This expanded list minimizes the occurrence of false-negative cases. **h**, Receiver operating characteristic (ROC) analysis for over-seg correction, comparing CS-Net correction results to a simple feature change score. CS-Net achieves superior discrimination (AUC = 0.96). **i**, Median trajectory length improvement (%) after applying correction, plotted against the maximum gap size (frames) in trajectory linking. **j**, Number of trajectories classified as corrected (green) or unchanged (gray) for each maximum gap size; each bar represents the total number of trajectories, not only candidate cases. **k**, Vacancy rate (%) of trajectories before (gray) and after (green) correction, shown across different maximum gap sizes.

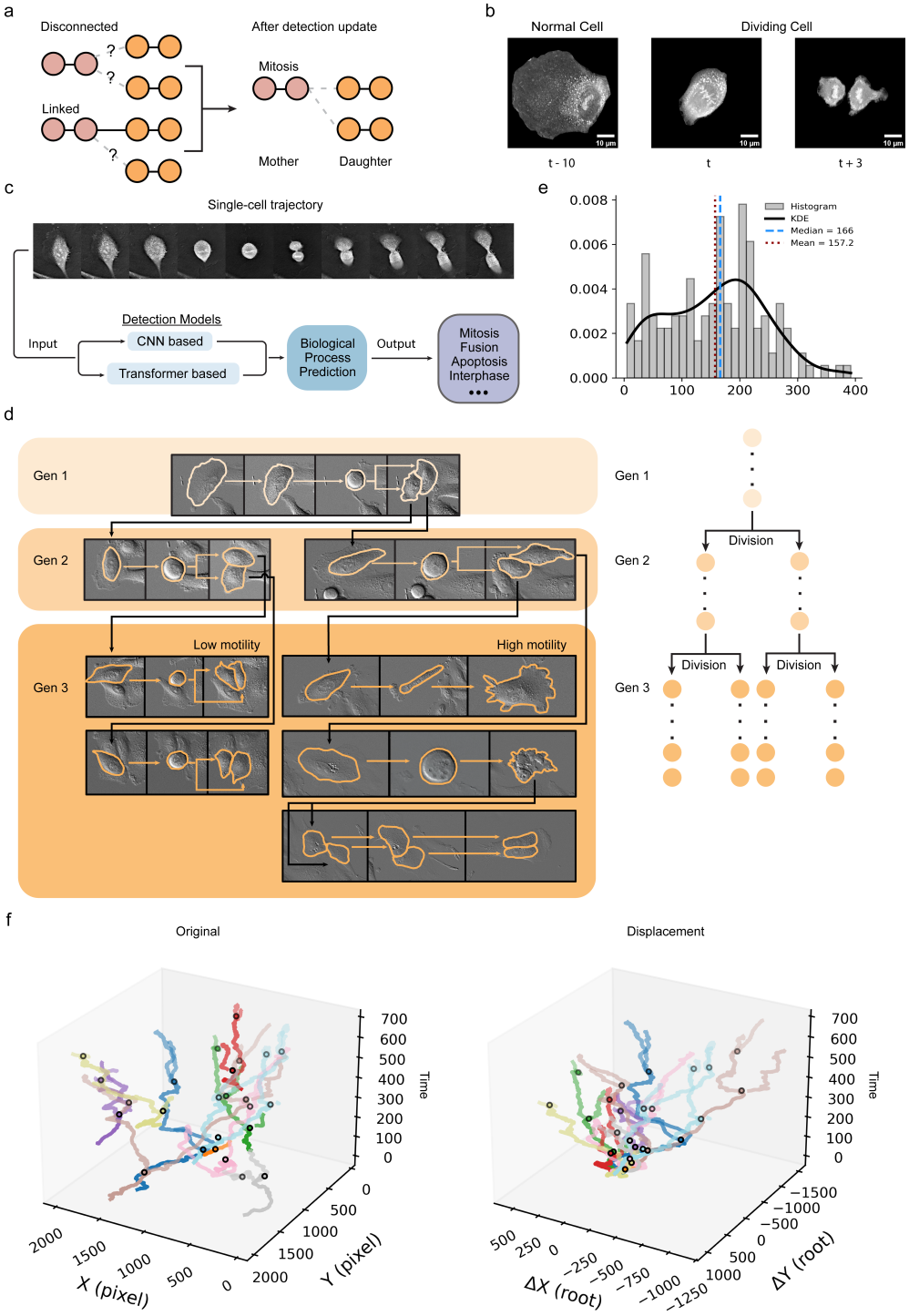

Fig. 4 Automated cell lineage reconstruction and biological process detection in LivecellX. a, Solution of linkage errors caused by mitosis events. In standard tracking, mitosis can lead to disconnected or ambiguous associations between mother and daughter cells. LivecellX detects mitosis events and accurately establishes lineage linkages between mother and both daughter cells. **b**, Distinctive appearances of cells before and after cell division in one single cell trajectory. **c**, Schematic of the biological process detection module. Single-cell trajectories are analyzed by CNN-based or transformer-based models to predict mitosis and other biological events, enabling automated lineage tree construction. **d**, Representative reconstructed cell lineage across three generations, visualized with cell contours in microscopy images. Generation 3 cells are further categorized by low and high motility, highlighting heterogeneity in movement within lineages. **e**, Distribution of cell cycle durations for reconstructed lineages, shown as a histogram with kernel density estimation (KDE). Mean and median durations are indicated. **f**, Three-dimensional visualization of cell lineage trajectories over time, shown as absolute coordinates (left) and as displacements from the root cell (right), revealing both lineage structure and motility patterns.

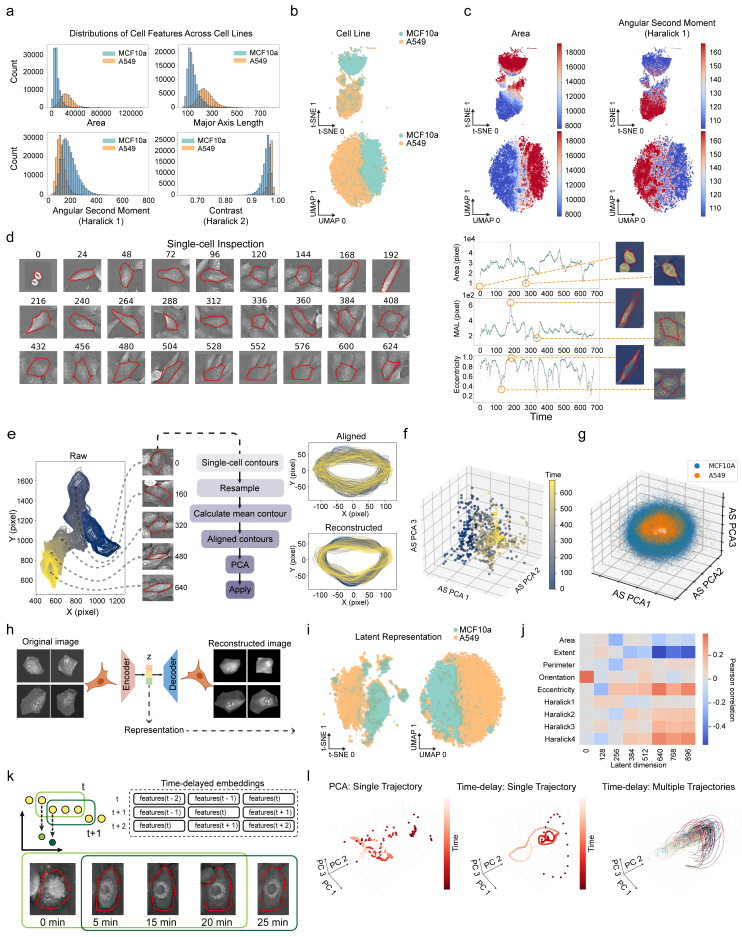

Fig. 5 Automated extraction and analysis of single-cell dynamics in LivecellX. a, Distributions of morphological (area, major axis length) and Haralick texture features (angular second moment, contrast) of A549 and MCF10A cells. **b** Projections of single-cell morphology and texture features using t-SNE (top) and UMAP (bottom). Colors represent cell lines. **c**, Projections of single-cell morphology and texture features using t-SNE (top) and UMAP (bottom). Colors represent area (left) and angular second moment(right). **d**, Temporal variation of morphological features (area, MAL, and eccentricity) along a single A549 cell trajectory. Left: sequential cell images and outlines. Right: feature dynamics and cell images at extreme feature values. **e**, Pipeline of AS model for quantifying dynamic cell contours. Left: single-cell contours colored by time. Middle: schematic workflow of AS model. Right: aligned (top) and PCA-reconstructed (bottom) contours of a single trajectory. **f**, Trajectory dynamics projected onto the first three PCs of the AS model, colored by time. **g**, PCA projection of AS model representations for A549 and MCF10A cells, revealing cell line–specific characteristics. **h**, Schematic of the Harmony deep learning model for extracting latent embeddings from single-cell images, with example reconstructed images. **i**, Visualization of Harmony latent representations with t-SNE and UMAP, colored by cell line. **j**, Correlations between Harmony latent features and traditional morphological or texture features, displayed as a heatmap. **k**, Illustration of time-delayed embedding applied to single-cell trajectories. Example cell images and masks (red outlines) are shown at different time points. **l**, Time delay embedding of cell cycle trajectory. Left: One cell cycle trajectory in the Harmony embedding feature space. Middle: Time delay embedding of the trajectory on the left panel. Right: Multiple time-delay embedding cell cycle trajectories. PCA is applied for visualization.

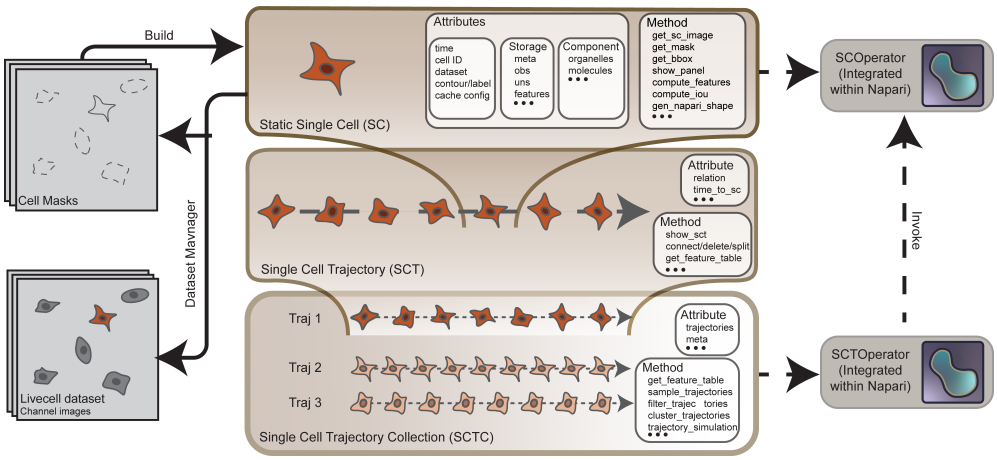

Fig. 6 Object-oriented architecture and user interface integration of LivecellX. Hierarchical data structures for representing static single cells (SC), single-cell trajectories (SCT), and collections of trajectories (SCTC). Key attributes and methods enable flexible querying, feature computation, and Napari-based interaction via SCOperator and SCTOperator modules.

5 Methods

5.1 Experimental data collection and live-cell imaging setup

MCF10A cells: Human breast epithelial MCF10A cells (ATCC, CRL-10317) were cultured in DMEM/F-12 medium supplemented with 5% horse serum, 20 ng/mL epidermal growth factor (EGF), 0.01 mg/mL insulin, 500 ng/mL hydrocortisone, and 100 ng/mL cholera toxin. Cells were maintained at 37 °C in a humidified incubator with 5% CO_2 and used within 10 passages. Holo-tomographic (HT) label-free images were acquired with an Nanolive CX-A microscope (Model: 3D Cell Explorer Automated, Part Number: CX03). For each experiment, 500–5,000 cells were seeded into 35 mm glass-bottom imaging dishes (Ibidi, 80137) two days prior to imaging. Cells under the microscope were maintained at 37 °C with 5% CO_2 using a Tokai Hit stage top incubator. HT images were acquired every 5 minutes, using a 60× objective (N.A. = 1.45). Imaging was performed over 57 hours, resulting in 684 frames. For experiments lasting over 48 hours, 300 µL of fresh medium was added every 48 hours.

A549 cells: Human lung adenocarcinoma A549 cells (ATCC, CCL-185) were cultured in F-12K medium supplemented with 10% fetal bovine serum (FBS) and maintained under the same incubation conditions (37 °C, 5% CO_2). Cells within 10 passages were used. Nanolive CX-A live-cell imaging was carried out with the same setup as described above. Imaging was conducted for 27 hours at 5-minute intervals, yielding 323 frames.

A549/VIM-RFP cells: A549 cells stably expressing Vimentin-RFP (ATCC, CCL-185EMT) were cultured in F-12K medium (Corning) supplemented with 10% FBS. Cells were seeded into 35 mm glass-bottom culture dishes (Cellvis D35-20-1.5N) and maintained at 37 °C in a humidified atmosphere with 5% CO_2. Cells from passages 3–10 were used for live-cell imaging with an Nikon Ti2-E microscope. For each experiment, 5,000–30,000 cells were plated two days before imaging. Differential interference contrast (DIC) images were acquired every 5 minutes, and TRITC channel images (Ex 555 nm / Em 587 nm) every 30 minutes using a 20× objective (N.A. = 0.75). Imaging was performed for 24 to 120 hours. For courses exceeding 48 hours, 500 µL of fresh culture medium was added every 48 hours.

5.2 Corrective Segmentation Network (CS-Net)

CS-Net is designed to refine erroneous segmentation masks by operating on cropped ROIs where single-cell segmentation errors are detected. It is implemented as a modular architecture using PyTorch Lightning and supports interchangeable backbones.

In its default configuration, it employs a DeepLabV3-ResNet50 backbone pretrained on ImageNet, with the final classifier replaced by a 1×1 convolutional layer to produce multi-class output. An auxiliary classifier is also attached to intermediate backbone features to predict categorical segmentation error types (e.g., over- or under-segmentation).

Input

For each identified error region, an ROI including the erroneous mask and its surrounding context is cropped from the original image and mask. Input tensors are shaped as $3 \times H \times W$, where H and W denote the height and width of the cropped ROI. The first channel contains the raw image. The second channel is a derived feature map (e.g., binary mask, label mask, Euclidean distance transform (EDT), or normalized EDT) as described in Section 5.2.2. The third channel duplicates the second to meet the three-channel requirement of pretrained backbones.

Output

CS-Net produces three output masks:

- **Segmentation mask** – A probability map representing the likelihood of each pixel belonging to a true cell object.
- **Under-seg mask** – A probability map indicating the likelihood of pixels lying outside the true object boundary but erroneously included in the original cell mask.
- **Over-seg mask** – A probability map indicating the likelihood of pixels that should be included in the cell mask but were mistakenly excluded.

Loss functions

The network supports cross-entropy (CE), mean squared error (MSE), and binary cross-entropy (BCE) loss functions, selected based on input types. While class and pixel-wise weighting can be applied during training, we found little performance difference with or without them. For model training, we use BCE as the loss function, specifically tailored to our requirements. Because each pixel may belong to multiple channels, it is inappropriate to use the standard cross-entropy across all classes. Instead, we compute the MSE alongside BCE, incorporating EDT to facilitate comparative evaluation.

During ablation tests, we explored various strategies for augmenting the other two channels. One approach involved removing backgrounds from raw images to create background-free representations of single cells in one of both channels. Additionally, we examined the effectiveness of incorporating segmentation masks and single-cell masks as two additional channels. This methodology enables us to assess the impact of these extra channels on the model's performance.

5.2.1 CS-Net evaluation metrics

Average Precision (AP) for object detection is typically computed using IoU-based matching; historically, *PASCAL VOC* reports AP for **bounding-box** detections at a fixed IoU threshold of 0.5 [57]. By contrast, *COCO* averages AP across multiple IoU thresholds ($0.50 - 0.95$ in steps of 0.05) and publishes metrics for both **bounding boxes** and **instance masks** (often denoted `AP_bbox` and `AP_mask`) [58, 59]. Bounding box-based metrics are often insufficient for detecting biologically relevant segmentation errors, as they cannot distinguish between closely packed or overlapping cells. Cell Match represents the percentage of ground-truth cells that are correctly paired with predicted cells, providing a direct measure of identification accuracy at the single-cell level. Correction success rate reports the proportion of samples in which every ground-truth cell is matched to exactly one predicted cell, serving as a stringent criterion for correctness. Intersection over Union (IoU) quantifies the overlap between predicted and ground-truth cell masks; higher IoU values indicate more accurate segmentation. Root Mean Square Error (RMSE) of normalized EDT masks measures pixel-level

differences in shape between predicted and ground-truth masks, with lower values indicating greater similarity.

Cell match rate is defined as the percentage of predicted masks that accurately correspond to ground-truth cells. Mathematically, let N represent the number of samples. For the i-th sample, let M_i denote the total number of ground-truth cells, and let $\hat{M}_i$ represent the number of predicted masks that correctly match a ground-truth cell based on a defined matching criterion. In our scenario, this criterion is IoU-based (IoU$\geq$0.5). Each ground-truth cell can be matched to at most one predicted cell. The cell match rate is then defined as:

$$\text{Cell Match Rate} = \frac{1}{N}\sum_{i=1}^{N}\frac{\hat{M}_i}{M_i}.$$

Correction success rate is a more stringent than the cell match rate. One segmentation error case typically relates with two or more masks. This metric requires that the predicted masks of CS-Net and ground-truth masks be *bijectively matched* within this case, meaning there must be a one-to-one correspondence between predicted masks and ground-truth masks. Each pair must satisfy a predefined matching criterion, which in our case is IoU is greater than 0.5, consistent with the criterion used in the cell match rate. A case is considered successfully corrected only if every ground-truth cell is accurately matched to a predicted cell, and vice versa. For the i-th case, let G_i represent the set of ground-truth masks, and P_i denote the set of predicted masks. Let $\mathcal{M}_i \subseteq G_i \times P_i$ be the set of matched pairs that satisfy the matching criterion. We define a successful correction if $|\mathcal{M}_i| = |G_i| = |P_i|$, i.e., a perfect one-to-one matching exists. Then, the correction success rate is given by:

$$\text{Correction Success Rate} = \frac{1}{N}\sum_{i=1}^{N}\mathbb{1}\left[|\mathcal{M}_i| = |G_i| = |P_i|\right],$$

where $\mathbb{1}[\cdot]$ is the indicator function, equal to 1 if the condition is true and 0 otherwise.

Root mean square error (RMSE) is a metric to quantify the difference between the predicted mask and the ground-truth mask. For a predicted mask M_{predict} and ground-truth mask M_{GT}, both represented as images of the same dimensions, RMSE is defined as:

$$\text{RMSE}(M_{\text{predict}}, M_{\text{GT}}) = \sqrt{\frac{1}{|M|}\sum_{x\in M}\left(M_{\text{predict}}(x) - M_{\text{GT}}(x)\right)^2},$$

where $|M|$ is the total number of pixels in the mask, and x indexes over all pixels. A lower RMSE signifies better alignment between predicted and ground-truth masks. RMSE is appropriate for evaluating both binary masks and EDT masks. However, when dealing with label masks, caution is advised, as the ground-truth and predicted labels may not align due to potential differences in label assignment.

5.2.2 Focus mechanism

To enhance the semantic awareness of the correction task, we introduce a *focus mechanism* that guides deep learning models to concentrate on a specific cell during a single forward pass. This mechanism is implemented by incorporating a dedicated focus channel, which supplies the model with the target cell mask generated in the segmentation step. Formally, given an input image tensor $X \in \mathbb{R}^{H\times W\times C}$ and a focus mask $F \in \mathbb{R}^{H\times W\times 1}$, we construct the focused input $\tilde{X}$ by concatenating F to X along the channel dimension:

$$\tilde{X} = \text{concat}(X, F),$$

where concat$(\cdot)$ denotes the channel-wise concatenation operation. The resulting tensor $\tilde{X} \in \mathbb{R}^{H\times W\times (C+1)}$ includes both the original image features and an additional semantic cue, thereby facilitating improved learning and inference.

Focus mask types

Different choices for F impact the effectiveness of the focus mechanism. We examined two primary strategies.

The first one is binary focus masks. A binary mask F highlights the target cell by setting the relevant pixels to 1 while marking all others as 0:

$$F_{i,j} = \begin{cases} 1, & \text{if } (i,j) \in \mathcal{C}_{\text{target}}, \\ 0, & \text{otherwise.} \end{cases}$$

Here, $\mathcal{C}_{\text{target}}$ denotes the region occupied by the target cell. One issue with this method is that the focus input mask for over-seg cases may be indistinguishable from correct masks, particularly in scenarios where two over-segmented pieces of a cell are joined together. A straightforward approach is to use label masks, where each integer label represents a distinct cell, instead of binary masks. However, since these label values lack semantic meaning, they can lead to training instability. In such instances, neither the binary focus mask nor the label focus mask prove to be effective. Inspired by recent advances in the field [8, 16], we explore the potential of EDT maps as F to resolve these issues.

The second one is Euclidean distance transform (EDT) maps. An EDT map F encodes the spatial relationship between pixels and the cell boundary:

$$F_{i,j} = \begin{cases} d_{i,j}, & \text{if } (i,j) \in \mathcal{C}_{\text{target}}, \\ 0, & \text{otherwise,} \end{cases}$$

where $d_{i,j}$ represents the Euclidean distance of pixel (i,j) to the closest boundary of the target cell. EDT maps offer a continuous gradient of focus intensity, helping to mitigate the instability issues often seen with label-based masks. Additionally, to prevent numerical and semantic problems arising from the variation in cell sizes, as illustrated in Fig. 2, we implement a normalization strategy to confine the EDT values of a cell within the range $[1, \text{range}_{\text{max}}]$: $d_{i,j}$ is normalized by first computing the scaling factor as

$$\text{factor} = \frac{\max(d_{i,j})}{\text{range}_{\text{max}} - 1},$$

and then transforming all nonzero values ($d_{i,j} \geq 1$) using

$$\tilde{d}_{i,j} = \frac{d_{i,j}}{\text{factor}} + 1.$$

This approach ensures that the normalized EDT values are scaled relative to the maximum distance, with a minimum value of 1 for positive distances (object regions). This scaling preserves numerical stability and retains semantic meaning for both the target cell and the background regions, while also creating a numerical gap between the background and the cell boundary (Fig. 2).

5.2.3 Training and testing for CS-Net

Data annotation

We annotated under-seg, over-seg and negative samples from the segmentation results of two models: (1) a pretrained `cyto2` model from Cellpose [8] fine-tuned on our holotomography microscopy data, and (2) a custom U-Net trained on DIC images from a previous study [16]. For the Cellpose model results, we used the human-in-the-loop interface to fine-tune on annotations from 15 time frames of the A549 dataset and 35 time frames of the MCF10A dataset. The fine-tuned model was then applied to all video frames to generate segmentation masks. We created datasets via two annotation protocols: (1) manual contour drawing after visual inspection of all masks, and (2) automated computation of IoMin or IoU scores between frames t and $t+1$, followed by expert annotation of triplet candidates representing possible under-seg or over-seg

errors. The first method is labor-intensive but yields precise annotations, while the second enables experts to label samples with just two clicks by weakly propagating masks from neighboring time points, trading precision for efficiency.

Erosion and dilation

To enhance CS-Net's robustness against segmentation errors, we apply erosion for over-seg and dilation for under-seg (Extended Data Fig.2).

The dilation and erosion of a specific pixel point in a binary mask are defined as follows:

$$\texttt{dilate}(M, x, y) = \max_{(i,j)\in K} M(x+i, y+j),$$
$$\texttt{erode}(M, x, y) = \min_{(i,j)\in K} M(x+i, y+j),$$

where M denotes a binary mask. K is the kernel (typically a 3×3 square). The functions $\texttt{dilate}(M, x, y)$ and $\texttt{erode}(M, x, y)$ return the dilated or eroded value of the mask M at pixel location (x, y). Dilation expands foreground regions by taking the local maximum over the kernel neighborhood, while erosion contracts them by taking the local minimum. In practice, we erode over-seg masks ($M \leftarrow \texttt{erode}(M)$) to emphasize disconnected boundaries and make the over-segmentation more pronounced. Conversely, we dilate under-seg masks ($M \leftarrow \texttt{dilate}(M)$) to further amplify the effects of merged regions. These augmentations simulate varying degrees of segmentation errors, encouraging CS-Net to generalize better. We avoid dilating over-seg masks or eroding under-seg masks, as these transformations may convert them into correct segmentations or another type of segmentation errors.

Varying padding sizes

Typically, an under-seg sample includes pixels from multiple adjacent cells, whereas over-seg or over-seg dropout cases may exclude parts of a cell. Since CS-Net does not have access to the true error category of an incorrect mask during inference, selecting an appropriate padding size for the cropped region of interest is essential to provide adequate information for correction.

Let M denote the bounding box of a single-cell mask, defined by its coordinates $(x_{\min}, y_{\min}, x_{\max}, y_{\max})$. We construct a padded ROI M^{pad} by extending the bounding box uniformly in all directions:

$$M^{\text{pad}} = \{(x, y) \mid x_{\min} - p \leq x \leq x_{\max} + p,\ y_{\min} - p \leq y \leq y_{\max} + p\},$$

where p is the padding margin in pixels. To evaluate CS-Net's robustness to contextual variation, we use multiple padding values $p \in \{20, 50, 80, 100\}$ when preparing training and testing datasets. In applications, we use a fixed padding size of $p = 50$ pixels, selected based on microscope magnification and the typical pixel-scale size of individual cells.

We define p as a configurable hyperparameter that can be adjusted based on specific application requirements. All padding is extracted directly from the original label-free imaging data, rather than using zero-padding, to preserve spatial context and relevant background information.

Training data augmentation

To improve model generalization, we apply a set of spatial and photometric augmentations to both images and corresponding masks. These include random horizontal and vertical flips, affine transformations (rotation, translation, scaling, and shear), and optional Gaussian blur. All inputs are resized to 256×256 pixels, with bilinear interpolation for images and nearest-neighbor interpolation for masks. Augmentations are applied consistently across channels to preserve spatial alignment. The EDT channel is normalized after transformation. Normalizing images from different microscopy modalities enforces consistent intensity distributions across datasets by aligning their means and standard deviations.

Post-processing

For our best-performing models trained with the normalized EDT objective, we generated binary segmentation masks by thresholding the normalized EDT at values greater than 1. The resulting predicted masks were post-processed using OpenCV. Specifically, we extracted contours with `cv2.findContours`, using `mode=cv2.RETR_EXTERNAL` and `method=cv2.CHAIN_APPROX_SIMPLE` to efficiently identify external cell boundaries.

Training and testing dataset composition

We assembled a comprehensive dataset for training and testing the CS-Net correction model using both holo-tomography and DIC microscopy data. The training set includes under-seg, over-seg, and negative samples, encompassing both real and synthetic errors. Specifically, HT data contributed 4,150 under-seg samples, 55 over-seg samples, 14,753 synthetic over-seg cases, 39,500 synthetic dropout-overseg samples, and 416 correct samples, with corresponding test sets comprising 1,095, 15, 3,992, 10,612, and 150 samples, respectively. The DIC training set includes diverse imaging conditions and sources, with real and synthetic under- and over-seg samples, as well as 4,084 correct examples; corresponding test sets contain 1,021 correct samples and a subset of real and synthetic segmentation errors.

5.3 Trajectory correction algorithms

In live-cell imaging, there are three major typical linkage distortions between single-cell masks caused by segmentation errors:

1. Under-seg of two cells from different trajectories: When two cells, one from trajectory A and the other from trajectory B, are under-segmented at time t, they are merged into a single mask and mistakenly assigned to the same trajectory. If the merged object is assigned to trajectory A, trajectory B typically loses a time point. In some cases, another nearby cell at time t may be incorrectly assigned to trajectory B instead (Fig. 3a). This type of wrong linkage can propagate, leading to a cascade of incorrect linkages during tracking, particularly those involving neighboring cells.

2. Over-seg of a single cell: If a cell at time t in a trajectory is segmented into multiple fragments, only one fragment is associated with the trajectory. The situation constitutes a typical over-seg case. Unlike under-seg, over-seg errors do not often result in direct tracking linkage failures. However, they can still degrade the quality of downstream trajectory analysis. In rare instances where the segmentation model frequently produces over-seg errors, the unassigned fragments may interfere with the tracking of neighboring cells, potentially leading to incorrect trajectory linkages (Fig. 3b).

3. Absent cell: When the segmentation algorithm fails to detect an existing cell at time t, no mask is produced for that location. As a result, the tracking algorithm may terminate the trajectory prematurely or mis-assign the cell in subsequent frames. This type of error can also cause confusion in nearby trajectories if the algorithm incorrectly links neighboring cells to compensate for the missing one (Fig. 3c).

To efficiently address tracking and segmentation errors in large-scale live-cell imaging, we developed an iterative trajectory correction algorithm that integrates spatial-temporal consistency checks with CS-Net (Fig. 3e). The method is parameterized by values including a temporal search window (3 frames), correction crop padding (20 pixels), a maximum number of correction iterations (default 100), and an exclusion zone for boundary cells (50 pixels). All parameters are tunable.

For under-seg, the algorithm inspects each trajectory for missing frames. When a gap is found, it examines whether any cell masks present in the missing frame overlap substantially with the trajectory's masks in adjacent frames, using an IoMin/IoU threshold=0.5. If such matches exist, the candidate error region is cropped and passed to the trained CS-Net, which classifies and corrects the segmentation. The resulting corrected masks are assigned back to the trajectories using the linear assignment problem (LAP) based on spatial relation. If the correction is successful, the trajectories are updated accordingly: splitting, merging, or extending as needed.

For over-seg, the criterion we used in the reported analyses is based on morphological dynamics. Here, the algorithm monitors the area of each segmented cell mask

over time and flag instances where the area changes abruptly beyond a predefined threshold. If such a candidate is detected, the local region is again input to CS-Net for correction and classification. For over-seg, typically there is only one trajectory involved, so the corrected single-cell mask is inserted directly back into the trajectory without any assignment step.

The entire process is repeated iteratively. After each global correction iteration, the trajectories are updated, and new candidate errors are detected based on the most recent corrections. Iterations continue until no further correction is possible or the maximum number of rounds is reached. In all cases, if a correction cannot be confidently corrected, the related linkages are left unchanged and the error is flagged, but not retried unless its context is modified in subsequent rounds. Users can utilize LivecellX GUI to visually inspect these samples and perform manual correction.

In tracking algorithm like SORT tracker [47], the maximum age parameter (max gap in frames) defines the largest number of consecutive frames a trajectory can persist without a successful linkage. Hence the impact of this parameter on the correction process should be evaluated.

The number of over-seg cases in our HT microscopy dataset is significantly low compared to under-seg cases. To address this imbalance, we synthesized 5,000 over-seg samples using the same protocol employed for generating synthetic over-seg training data. We evaluated our trajectory correction algorithms on both the synthetic and real over-seg datasets.

5.3.1 Trajectory correction evaluation metrics

We introduce **trajectory vacancy** as a straightforward yet informative metric for evaluating the quality of a single-cell trajectory. For a given trajectory, the vacancy rate quantifies the extent to which it is fragmented over time. Specifically, we define the trajectory vacancy rate as follows::

$$\text{Trajectory Vacancy Rate} = 1 - \frac{\hat{n}}{t_{\max} - t_{\min} + 1}$$

where $t_{\max}$ and $t_{\min}$ are the maximum and minimum time frames of the cells in the trajectory after tracking, and $\hat{n}$ is the number of tracked cells in the trajectory. A lower vacancy rate indicates a more complete and continuous trajectory.

5.4 Biological process detection and lineage reconstruction

LivecellX supports automated classification of single-cell trajectories derived from temporal features and time-lapse image sequences. Let each single cell trajectory be represented as a fixed-length sequence of feature vectors:

$$\mathbf{x}_i = \left[x_i^{t_0}, x_i^{t_1}, \ldots, x_i^{t_{T-1}}\right], \quad x_i^t \in \mathbb{R}^d,$$

where t_0 is centered around the cell's birth event and T (typically ≈ 8) defines the length of the sliding window sampling temporal context. Each trajectory $\mathbf{x}_i$ belongs to a biological process label $y_i \in \{1, \ldots, C\}$, indicating its class over the entire sequence. The goal is to train a model f_θ that maps whole trajectories to their corresponding classes:

$$f_\theta \colon \mathbb{R}^{T \times d} \to \{1, \ldots, C\}, \quad \hat{y}_i = f_\theta(\mathbf{x}_i).$$

This formulation ensures classification captures dynamic cellular behaviors rather than static snapshots.

LivecellX facilitates the preparation of machine learning–ready inputs by providing helper functions that format temporal features, image patches, and video sequences directly from its data structures (e.g., single-cell and trajectory objects). The modular detection pipeline is extensible, enabling users to plug in custom classifiers f_θ. We provides pretrained CNN- and transformer-based models optimized for our DIC and HT A549 datasets produced in our study.

Lineage reconstruction in LivecellX is carried out after detecting biological processes. The framework provides utility functions to split or merge trajectories based on common cellular events, such as mitosis and cell fusion. In the case of mitosis, the mother and daughter trajectories are separated at the time point of division. For cell fusion, two individual cell trajectories are combined into a single trajectory at the fusion time point. The relationships between cell trajectories are stored within the LivecellX data structures. Users have the flexibility to define new types of relationships either programmatically or through the graphical user interface.

To facilitate lineage visualization, LivecellX provides a light-weight tree traversal utility based on depth-first search (DFS). Within this algorithm, each trajectory is represented as a node in the lineage tree, and mother-daughter relationships are managed via the trajectory object structure. Starting from a root cell trajectory, the DFS-based algorithm recursively visits descendants, generating ordered lineage sequences or visual renderings.

5.5 Feature extraction and trajectory dynamics

The module performs feature extraction using the active shape model, Haralick features, local binary pattern, variational auto-encoder, Harmony [51], and diffusion models [60, 61]. LivecellX integrates the active shape model implemented in Celltool [32]. Haralick features and local binary patterns are extracted using the Mahotas package [62]. The variational auto-encoder is implemented through Harmony. LivecellX provides helper functions for constructing time delay embeddings from trajectories.

Harmony VAE for Microscopy Image Disentanglement

We adopted the original Harmony framework [51], a generic VAE-based model designed to disentangle semantic content from parameterized transformations in an unsupervised manner. Harmony combines a standard encoder–decoder architecture with a cross-contrastive learning objective to separate transformation-specific and content-specific latent factors. In its original form, Harmony demonstrated strong performance on structured datasets such as MNIST and cryo-EM images. However, we found that this architecture does not generalize effectively for our label-free microscopy datasets, where high noise levels and fine-scale cellular heterogeneity pose additional challenges. A primary limitation of the original model in our setting was posterior collapse, where the learned latent variables failed to capture meaningful variation and reconstructions became nearly identical regardless of latent input [8]. Additionally, the model often produced nearly identical reconstructions across different latent codes, indicating poor disentanglement and underfitting of the semantic space. To address these shortcomings, we replaced Harmony's original fully connected encoder and decoder with convolutional architectures tailored for high-resolution microscopy images. Specifically, we employed a deeper convolutional encoder with residual blocks and batch normalization to extract hierarchical visual features, and a mirrored transposed convolutional decoder to enhance spatial resolution during reconstruction. These architectural changes substantially increased the model capacity and improved its ability to capture both global cell shape and fine-grained morphological variations. We retained Harmony's cross-contrastive loss structure, which includes three dissimilarity terms—between the image reconstructed from semantic latent space and the image transformed by the predicted transformation parameters, between the transformed outputs of original and augmented views, and between the reconstruction from the augmented input and its transformed version—along with a KL divergence term enforcing proximity of semantic latent distributions between similar inputs. These components jointly drive the disentanglement of latent representations while avoiding trivial solutions. In our implementation, we further regularized transformation parameters through appropriate activation functions and shared kernel weights. We empirically found that our convolutional Harmony variant better disentangles latent content from cell shape variability and imaging transformations. This enhanced formulation enables generation of synthetic microscopy images and improves robustness to transformation-induced noise.

Harmony VAE: Original Formulation and Our Adaptations for Label-free Microscopy Images

The Harmony model aims to disentangle semantic content $z \in \mathbb{R}^d$ from transformation parameters $k \in \mathbb{R}^m$ using a variational autoencoder framework combined with a contrastive learning objective.

Let $x \in \mathcal{X}$ be an input image. The encoder maps x to both transformation parameters and a variational distribution over semantic latent variables:

$$(k, \psi) = f_\theta(x), \quad \psi = (\mu, \sigma) \Rightarrow z \sim \mathcal{N}(\mu, \mathrm{diag}(\sigma^2)).$$

The decoder reconstructs from the latent code, while the transformation module synthesizes a transformed image:

$$x_z = g_\phi(z), \quad x_k = T(x, k),$$

where $T(\cdot, k)$ denotes a differentiable transformation function parameterized by k.

Harmony's training objective is defined as:

$$\mathcal{L}(x, x') = \gamma \left[D_1(x_z, x_k) + D_2(x_k, x''_{k'}) + D_3(x''_{k'}, x''_{z'}) \right] + \mathrm{KL}\left(P_\psi \| P_{\psi'} \right),$$

where $x' = T(x, k')$ is a randomly transformed view, and $x''_{k'}, x''_{z'}$ are outputs from applying the encoder and decoder to x'. Each D_i denotes a dissimilarity function, typically mean squared error (MSE).

Modifications on Harmony

To adapt Harmony for high-resolution and morphologically complex microscopy data, we replaced the fully connected encoder–decoder architecture with CNNs to better capture spatial structure.

Modified Encoder: Instead of the original MLP-based encoder,

$$f_\theta(x) = \mathrm{MLP}(x),$$

we use a convolutional encoder:

$$f_\theta^{\mathrm{conv}}(x) = \mathrm{CNN}_{\mathrm{enc}}(x) = \left(\mu(x), \sigma(x), k(x) \right),$$

where $\mu(x)$ and $\sigma(x)$ are obtained from global average pooled convolutional features followed by linear projections.

Modified Decoder: Replacing the MLP decoder,

$$g_\phi(z) = \mathrm{MLP}(z),$$

we use a convolutional decoder:

$$g_\phi^{\mathrm{conv}}(z) = \mathrm{CNN}_{\mathrm{dec}}(z),$$

consisting of upsampling and convolutional blocks to reconstruct high-resolution images.

Updated Forward Pass: The modified reconstruction path becomes:

$$z \sim \mathcal{N}(\mu(x), \mathrm{diag}(\sigma(x)^2)), \quad x_z = \mathrm{CNN}_{\mathrm{dec}}(z), \quad x_k = T(x, k(x)).$$

This convolutional formulation enhances Harmony's representational capacity, improves robustness to complex transformations, and effectively mitigates posterior collapse in microscopy imaging applications.

6 Data Availability

Raw live-cell label-free images are stored and shared on OSF.

7 Code Availability

The `LivecellX` package is publicly available at: https://github.com/xing-lab-pitt/livecellx. The repository includes source code, installation instructions, and example notebooks to facilitate reproduction and further development.

Acknowledgements. This work was partially supported by National Science Foundation (MCB2205148) to JX, National Institute of Biomedical Imaging and Bioengineering (T32EB009403) to DP, National Natural Science Foundation of China Grants (No.12247104) to WW and the University of Pittsburgh Center for Research Computing and Data, `SCR_022735` through the resources provided. Specifically, this work used the H2P cluster, which is supported by NSF award number (OAC-2117681).

Declarations

The authors declare no conflict of interest.

References

[1] Berge, K.V.D., Bézieux, H.R., Street, K., Saelens, W., Clement, L.: Trajectory-based differential expression analysis for single-cell sequencing data. Nature Communications **11**(1) (2020)

[2] Schiebinger, G., Shu, J., Tabaka, M., Cleary, B.L., Lander, E.S.: Optimal-transport analysis of single-cell gene expression identifies developmental trajectories in reprogramming. Cell **176**(6), 1517 (2019)

[3] Saelens, W., Cannoodt, R., Todorov, H., Saeys, Y.: A comparison of single-cell trajectory inference methods. Nature Biotechnology (2019)

[4] Manno, G.L., Soldatov, R., Zeisel, A., Braun, E., Hochgerner, H., Petukhov, V., Lidschreiber, K., Kastriti, M.E., Lnnerberg, P., Furlan, A.: Rna velocity of single cells. Nature (2018)

[5] Qiu, X., Zhang, Y., Martin-Rufino, J.D., Weng, C., Hosseinzadeh, S., Yang, D., Pogson, A.N., Hein, M.Y., Min, K.H.J., Wang, L.: Mapping transcriptomic vector fields of single cells. Cell (4), 185 (2022)

[6] Chen, Z., King, W.C., Hwang, J. Mark Zhang: Deepvelo: Single-cell transcriptomic deep velocity field learning with neural ordinary differential equations. science advances **8**(48) (2022)

[7] Bray, M.A., Singh, S., Han, H., Davis, C.T., Borgeson, B., Hartland, C., Kost-Alimova, M., Gustafsdottir, S.M., Gibson, C.C., Carpenter, A.E.: Cell painting, a high-content image-based assay for morphological profiling using multiplexed fluorescent dyes. Nature Protocols **11**(9), 1757–1774 (2016)

[8] Stringer, C., Wang, T., Michaelos, M., Pachitariu, M.: Cellpose: a generalist algorithm for cellular segmentation. Nature Methods **18**, 100–106 (2021) https://doi.org/10.1038/s41592-020-01018-x

[9] Gordonov, S., Hwang, M.K., Wells, A., Gertler, F.B., Lauffenburger, D.A., Bathe, M.: Time series modeling of live-cell shape dynamics for image-based phenotypic profiling. Integr Biol (Camb) **8**(1), 73–90 (2016) https://doi.org/10.1039/C5IB00239A

[10] Wang, W., Douglas, D., Zhang, J., Kumari, S., Enuameh, M.S., Dai, Y., al.: Live-cell imaging and analysis reveal cell phenotypic transition dynamics inherently missing in snapshot data. Science Advances **6**(36), 9319 (2020) https://doi.org/10.1126/sciadv.aba9319

[11] Skylaki, S., Hilsenbeck, O., Schroeder, T.: Challenges in long-term imaging and quantification of single-cell dynamics. Nature Biotechnology **34**(11), 1137–1144 (2016)

[12] Specht, E.A., Braselmann, E., Palmer, A.E.: A critical and comparative review of fluorescent tools for live-cell imaging. Annual Review of Physiology **79**(1), 93 (2017)

[13] Uchida, S.: Image processing and recognition for biological images. Dev Growth Differ **55**(4), 523–549 (2013) https://doi.org/10.1111/dgd.12033

[14] Shelhamer, E., Long, J., Darrell, T.: Fully convolutional networks for semantic segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence **39**(4), 640–651 (2017) https://doi.org/10.1109/TPAMI.2016.2572683

[15] Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: Medical Image Computing and Computer-Assisted Intervention – MICCAI, pp. 234–241. Springer, ??? (2015). https://doi.org/10.1007/978-3-319-24574-4_28

[16] Wang, W., Taft, D.A., Chen, Y.-J., Zhang, J., Wallace, C.T., Xu, M., al.: Learn to segment single cells with deep distance estimator and deep cell detector. Computers in Biology and Medicine **108**, 133–141 (2019) https://doi.org/10.1016/j.compbiomed.2019.03.011

[17] Van Valen, D.A., Kudo, T., Lane, K.M., Macklin, D.N., Quach, N.T., DeFelice, M.M., al.: Deep learning automates the quantitative analysis of individual cells in live-cell imaging experiments. PLOS Computational Biology **12**(11), 1005177 (2016) https://doi.org/10.1371/journal.pcbi.1005177

[18] Greenwald, N.F., Miller, G., Moen, E., Kong, A., Kagel, A., Dougherty, T., Fullaway, C.C., Mcintosh, B.J., Leow, K.X., Schwartz, M.S.: Whole-cell segmentation of tissue images with human-level performance using large-scale data annotation and deep learning. Nature biotechnology (4), 40 (2022)

[19] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. CoRR **abs/2010.11929** (2020) 2010.11929

[20] Wang, Y., Zhao, J., Xu, H., Han, C., Tao, Z., Zhou, D., Geng, T., Liu, D., Ji, Z.: A systematic evaluation of computational methods for cell segmentation. Briefings in Bioinformatics **25**(5), 407 (2024) https://doi.org/10.1093/bib/bbae407

[21] Maška, M., Ulman, V., Delgado-Rodriguez, P., Gómez-de-Mariscal, E., Nečasová, T., Guerrero Peña, F.A., Ren, T.I., Meyerowitz, E.M., Scherr, T., Löffler, K., Mikut, R., Guo, T., Wang, Y., Allebach, J.P., Bao, R., Al-Shakarji, N.M., Rahmon, G., Toubal, I.E., Palaniappan, K., Lux, F., Matula, P., Sugawara, K., Magnusson, K.E.G., Aho, L., Cohen, A.R., Arbelle, A., Ben-Haim, T., Riklin Raviv, T., Isensee, F., Jäger, P.F., Maier-Hein, K.H., Zhu, Y., Ederra, C., Urbiola, A., Meijering, E., Cunha, A., Muñoz-Barrutia, A., Kozubek, M., Ortiz-de-Solórzano, C.: The cell tracking challenge: 10 years of objective benchmarking. Nature Methods **20**(7), 1010–1020 (2023) https://doi.org/10.1038/s41592-023-01879-y

[22] Kuhn, H.W.: The hungarian method for the assignment problem. Naval Research Logistics Quarterly **2**(1-2), 83–97 (1955)

[23] Reid, D.B.: An algorithm for tracking multiple targets. IEEE Transactions on Automatic Control **24**(6), 843–854 (1979)

[24] Wojke, N., Bewley, A., Paulus, D.: Simple online and realtime tracking with a deep association metric. In: 2017 IEEE International Conference on Image Processing (ICIP), pp. 3645–3649 (2017). IEEE

[25] Bragantini, J., Theodoro, I., Zhao, X., Huijben, T.A.P.M., Hirata-Miyasaki, E., VijayKumar, S., Balasubramanian, A., Lao, T., Agrawal, R., Xiao, S., Lammerding, J., Mehta, S., Alexandre X. Falcão, A.J., Lange, M., Royer, L.A.: Ultrack: pushing the limits of cell tracking across biological scales. Nature Methods (2025)

[26] Moen, E., Borba, E., Miller, G., Schwartz, M., Bannon, D., Koe, N., al.: Accurate cell tracking and lineage construction in live-cell imaging experiments with deep learning. bioRxiv, 803205 (2019) https://doi.org/10.1101/803205

[27] Skylaki, S., Hilsenbeck, O., Schroeder, T.: Challenges in long-term imaging and

quantification of single-cell dynamics. Nature Biotechnology **34**(11), 1137–1144 (2016) https://doi.org/10.1038/nbt.3664

[28] Viana, M.P., Chen, J., Knijnenburg, T.A., Vasan, R., Yan, C., Arakaki, J.E., Bailey, M., Berry, B., Borensztejn, A., Brown, E.M.: Integrated intracellular organization and its variations in human ips cells. Nature **613** (2023)

[29] J., X.: Reconstructing data-driven governing equations for cell phenotypic transitions: integration of data science and systems biology. Phys Biol. **19(6)** (2022)

[30] Way, G.P., Natoli, T., Adeboye, A., Litichevskiy, L., Yang, A., Lu, X., al.: Morphology and gene expression profiling provide complementary information for mapping cell state. Cell Systems **13**(11), 911–9239 (2022) https://doi.org/10.1016/j.cels.2022.10.014

[31] Gut, G., Herrmann, M.D., Pelkmans, L.: Multiplexed protein maps link subcellular organization to cellular states. Science **361**(6401), 7173 (2018) https://doi.org/10.1126/science.aar7173

[32] Pincus, D., Pincus, S.: Comparison of quantitative methods for cell-shape analysis. Journal of Microscopy **227**(2), 140–156 (2007)

[33] Alizadeh, E., Castle, J., A, A.Q., Taylor, C.D.L., Xu, W., Prasad, A.: Cellular morphological features are predictive markers of cancer cell state - sciencedirect. Computers in Biology and Medicine **126** (2020)

[34] Wang, W., Poe, D., Yang, Y., Hyatt, T., Xing, J.: Epithelial-to-mesenchymal transition proceeds through directional destabilization of multidimensional attractor. eLife **11** (2021)

[35] Copperman, J., Gross, S.M., Chang, Y.H., *et al.*: Morphodynamical cell state description via live-cell imaging trajectory embedding. Communications Biology **6**, 484 (2023) https://doi.org/10.1038/s42003-023-04837-8

[36] Hu, T., Xu, S., Wei, L., Zhang, X., Wang, X.: Celltracker: an automated toolbox for single-cell segmentation and tracking of time-lapse microscopy images. Bioinformatics **37**(2), 285–287 (2021) https://doi.org/10.1093/bioinformatics/btaa835

[37] Amat, F., Lemon, W., Mossing, D.P., McDole, K., Wan, Y., Branson, K., al.: Fast, accurate reconstruction of cell lineages from large-scale fluorescence microscopy data. Nature Methods **11**(9), 951–958 (2014) https://doi.org/10.1038/nmeth.2985

[38] Padovani, F., Mairhörmann, B., Falter-Braun, P., Lengefeld, J., Schmoller, K.M.: Segmentation, tracking and cell cycle analysis of live-cell imaging data with cell-acdc. BMC Biology **20**(1), 174 (2022) https://doi.org/10.1186/s12915-022-01484-1

[39] Piltti, K.M., Cummings, B.J., Carta, K., Manughian-Peter, A., Worne, C.L., Singh, K., al.: Live-cell time-lapse imaging and single-cell tracking of in vitro cultured neural stem cells - tools for analyzing dynamics of cell cycle, migration, and lineage selection. Methods **133**, 81–90 (2018) https://doi.org/10.1016/j.ymeth.2017.12.003

[40] Tian, C., Yang, C., Spencer, S.L.: Elliptrack: A global-local cell-tracking pipeline for 2d fluorescence time-lapse microscopy. Cell Reports **32**(5), 107984 (2020) https://doi.org/10.1016/j.celrep.2020.107984

[41] Hernandez, D.E., Chen, S.W., Hunter, E.E., Steager, E.B., Kumar, V.: Cell tracking with deep learning and the viterbi algorithm. In: 2018 International Conference on Manipulation, Automation and Robotics at Small Scales (MARSS), pp. 1–7 (2018)

[42] napari contributors: napari: a multi-dimensional image viewer for python. https://doi.org/10.5281/zenodo.3555620 (2019). https://doi.org/10.5281/zenodo.3555620

[43] Design, V.S.: Understanding oversegmentation and region merging. "Oversegmentation is the process by which the objects ... are themselves segmented or fractured into subcomponents." (1998)

[44] Eggleston, P.: Understanding undersegmentation and region splitting (1999)

[45] Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: From contours to regions: An empirical evaluation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2011). Defines over- and under-segmentation in hierarchical segmentation context

[46] Felzenszwalb, P.F., Huttenlocher, D.P.: Distance transforms of sampled functions. Theory of Computing **1**(1), 1–48 (2004)

[47] Upcroft, A., Zaga, G., La, O., Fry, B.: Simple online and realtime tracking (2016)

[48] Ulicna, K., Vallardi, G., Charras, G., Lowe, A.R.: Automated deep lineage tree analysis using a bayesian single cell tracking approach. Frontiers in Computer Science **3**, 92 (2021) https://doi.org/10.3389/fcomp.2021.734559

[49] Bove, A., Gradeci, D., Fujita, Y., Ban erjee, S., Charras, G., Lowe, A.R.: Local cellular neighborhood controls proliferation in cell competition. Molecular Biology of the Cell **28**(23), 3215–3228 (2017) https://doi.org/10.1091/mbc.E17-06-0368

[50] Zhao, Y., Xiong, Y., Lin, D.: MMAction: An open-source toolbox for action understanding. https://github.com/open-mmlab/mmaction (2019)

[51] Uddin, M.R., Howe, G., Zeng, X., Xu, M.: Harmony: A generic unsupervised approach for disentangling semantic content from parameterized transformations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 20614–20623 (2022). https://doi.org/10.1109/cvpr52688.2022.01999

[52] Takens, F.: Detecting strange attractors in turbulence. In: Rand, D., Young, L.-S. (eds.) Dynamical Systems and Turbulence, Warwick 1980, pp. 366–381. Springer, Berlin, Heidelberg (1981)

[53] Weigert, M., Schmidt, U., Haase, R., Sugawara, K., Myers, G.: Star-convex polyhedra for 3d object detection and segmentation in microscopy. In: The IEEE Winter Conference on Applications of Computer Vision (WACV) (2020). https://doi.org/10.1109/WACV45572.2020.9093435

[54] Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., Girshick, R.: Detectron2. https://github.com/facebookresearch/detectron2 (2019)

[55] Christiansen, E.M., Yang, S.J., Ando, D.M., Javaherian, A., Skibinski, G., Lipnick, S., al.: In silico labeling: Predicting fluorescent labels in unlabeled images. Cell **173**(3), 792–80319 (2018) https://doi.org/10.1016/j.cell.2018.03.047

[56] Ounkomol, C., Seshamani, S., Maleckar, M.M., Collman, F., Johnson, G.R.:

Label-free prediction of three-dimensional fluorescence images from transmitted-light microscopy. Nature Methods **15**(11), 917–920 (2018) https://doi.org/10.1038/s41592-018-0151-5

[57] Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. International Journal of Computer Vision **88**(2), 303–338 (2010) https://doi.org/10.1007/s11263-009-0275-4

[58] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European Conference on Computer Vision (ECCV), pp. 740–755 (2014)

[59] Detectron2 Contributors: COCO Evaluator in Detectron2 Documentation. https://detectron2.readthedocs.io/en/v0.5/_modules/detectron2/evaluation/coco_evaluation.html. Accessed 2025-10-14 (2020)

[60] Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. In: Advances in Neural Information Processing Systems (NeurIPS), vol. 33, pp. 6840–6851 (2020)

[61] Fick, A.: On liquid diffusion. Annalen der Physik **170**(1), 59–86 (1855)

[62] Coelho, L.P.: Mahotas: Open source software for scriptable computer vision. Journal of Open Research Software **1**(1), 3 (2013) https://doi.org/10.5334/jors.ac