Aardvark: Composite Visualizations of Trees, Time-Series, and Images

Devin Lange, Robert Judson-Torres, Thomas A. Zangle, and Alexander Lex

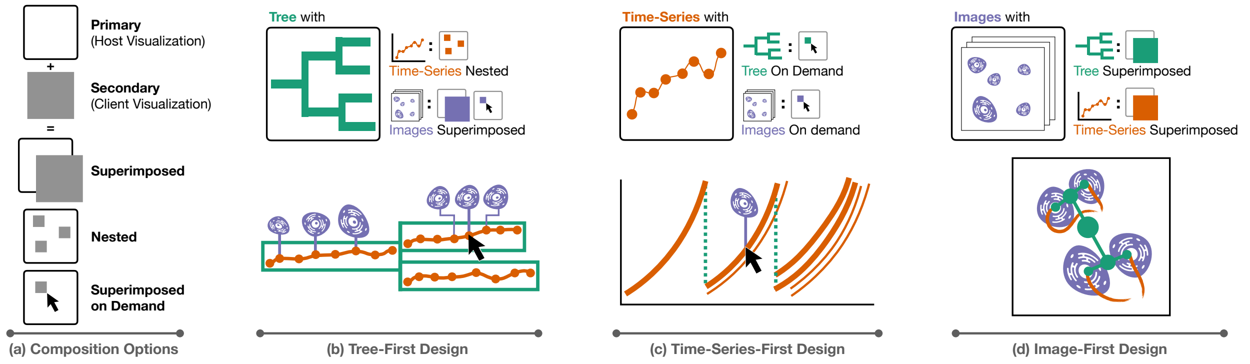

Fig. 1: Composition options and designs for cell microscopy visualizations. (a) Composition options for integrating visualizations in one view. First, a primary data type and visual encoding are chosen as the host visualization, and then additional data is added via composition as client visualizations. (b) The Tree-First Design uses a node-link diagram for the primary encoding (green), nests time-series in the nodes (orange), and superimposes cell image data at regular intervals and on demand (violet). (c) The Time-Series-First Design employs a line chart for the time-series data (orange) and superimposes topological data (green) and image data (violet) on demand. (d) The Image-First Design superimposes tree data and time-series data (cell movement) in the same coordinate system as the images.

Abstract—How do cancer cells grow, divide, proliferate, and die? How do drugs influence these processes? These are difficult questions that we can attempt to answer with a combination of time-series microscopy experiments, classification algorithms, and data visualization. However, collecting this type of data and applying algorithms to segment and track cells and construct lineages of proliferation is error-prone; and identifying the errors can be challenging since it often requires cross-checking multiple data types. Similarly, analyzing and communicating the results necessitates synthesizing different data types into a single narrative. State-of-the-art visualization methods for such data use independent line charts, tree diagrams, and images in separate views. However, this spatial separation requires the viewer of these charts to combine the relevant pieces of data in memory. To simplify this challenging task, we describe design principles for weaving cell images, time-series data, and tree data into a cohesive visualization. Our design principles are based on choosing a primary data type that drives the layout and integrates the other data types into that layout. We then introduce Aardvark, a system that uses these principles to implement novel visualization techniques. Based on Aardvark, we demonstrate the utility of each of these approaches for discovery, communication, and data debugging in a series of case studies.

Index Terms—Visualization, Cell Microscopy, View Composition.

1 Introduction

Time-series cell microscopy is an essential method to understand the behavior of cells in the life sciences and plays a large role in the study of cancer. One way to better understand this disease is to collect data with a microscope on cancer cells as they grow and divide. Microscopic **images** of the cells growing and dividing form the raw data of many derived analysis methods. Once cells are segmented to find their boundary and tracked over time, **time-series** data of derived cell attributes can describe how cells change over time. Finally, identifying cell divisions and recording the parent-child relationship produces an inheritance **tree** that describes the lineages of cells.

- *Devin Lange is with the University of Utah. E-mail: devin@sci.utah.edu*
- *Robert Judson-Torres is with the University of Utah. E-mail: Robert.Judson-Torres@hci.utah.edu*
- *Thomas A. Zangle is with the University of Utah. E-mail: tzangle@chemeng.utah.edu*
- *Alexander Lex is with the University of Utah. E-mail: alex@sci.utah.edu*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

Collecting such data can be challenging; it requires expertise in constructing and programming automated microscopes, physically preparing experiments in the lab, implementing machine learning and algorithms to perform image analysis, and analyzing the derived data. As with any complex data acquisition and processing pipeline, things can go wrong at any step. Consequently, it can be challenging to determine if and where errors exist, analyze the results, and communicate them.

For complex multimodal data, forming conclusions using one type of data can be difficult or impossible. For example, if a sudden decrease in cell size is observed, it is impossible to determine if that change represents a real biological phenomenon (the cell dying) or an error in the data (a segmentation failure) without reviewing the image data. An expert can often distinguish these possibilities when reviewing image data, but matching the observed effect in time-series data to the right cell in the right image is tedious with current methods. When constructing tree visualizations for lineage data, we run into a similar issue: matching nodes in the tree to time-series or image data is difficult.

There are two main contributions in this paper. First, we describe design principles for composite visualizations of images, time-series, and tree data by choosing a primary encoding and embedding the secondary data types within it. Next, we apply these design principles to our data (cell images, time-series data, and lineages) and implement it in an open-source visualization system we call Aardvark.[1] Aardvark includes three novel visualization designs that make the correspondence between data types easy to understand. We demonstrate the utility of this technique for data exploration, quality control, and communication with three case studies using cancer cell microscopy data.

2 Related Work

Our work is related to visualizations of cell microscopy data, multivariate trees, and imaging data. To our knowledge, no prior work combines all three of these modalities into one framework.

2.1 Cell Biology Visualizations

Many visualization tools are tailored to the life sciences. For instance, genomics [42, 43, 57], connectomics [1, 4, 71], and histology [27, 33, 68, 74] have utilized visualizations. Kerren and Schreiber argue that integrating diverse data modalities into a single visualization system is valuable [30]. One common data type from biology, imaging data, is well-suited for visualizations due to the visual nature of the data [73]. Visualizations of imaging data have been applied to subcellular structures [14, 70]. However, for our use case, we are interested in the analysis of whole cells, not subcellular structures. Visualizations for single-cell analysis can be utilized in different ways. Polyphony [13] uses visualizations to help researchers annotate cell types. This work differs from ours in that it does not include images of cells, and it focuses on cell-type annotation. However, it illustrates why fully automated approaches are insufficient for this domain. Single-cell analysis can also be applied to drug screening by measuring the differences in cell populations when exposed to different drugs. Screenit [19] is a visualization approach for doing this type of high-content screening. More general frameworks for analyzing and visualizing single-cell data are also available. CellProfiler [11] is a system for cell image analysis, and CellProfiler Analyst [28] extends the base system to support data exploration with visualizations . SpatialData [45] links images of cells and their positions to their derived attributes. Finally, Vitessce [29] provides multiple linked visualizations for exploring and analyzing single-cell data. However, none of these frameworks provide a visualization that combines time-series, trees, and imaging data.

Live cell microscopy tracks cell development under a microscope. The idea of visualizing these tracks has been around for at least two decades [18]. In the last decade, automation of data collection and tracking has become more prevalent [24, 47]. Our collaborators use commercial systems — such as Livecyte [2] and HoloMonitor [65] — as well as code developed within their labs. These systems use quantitative phase imaging (QPI) to measure the mass of individual cells [51]. Our previous work, Loon [35], visualized this QPI data; however, Loon does not consider lineages. Pretorius et al. have defined six classes of visualizations for live cell microscopy data [59]. Our approach utilizes three of these classes (spatial embedding, temporal plots, and lineage diagrams) along with the raw imaging data.

The most closely related live cell microscopy visualization systems focus on studying how individual cells grow into organisms or the process of embryogenesis. The mechanisms that drive how humans and other multi-cellular organisms grow and develop have fascinated scientists for centuries; visualizations have been part of this journey from early on and continue to this day [15, 56]. Meyer et al. developed MulteeSum [49] to compare the embryo development of fruit flies. However, cell lineages are not collected or visualized in this work; instead, the spatial relationship of cells is utilized. Still, domain scientists value visualization tools for showing the actual tree information of the cell lineage [12]. CeLaVi [61] is one such tool that links a lineage diagram with 3D cell positions. LineageD [25] is another tool that also supports editing cell lineage labels. However, both determine cell relationships from a snapshot in time, which differs from our work in that the full lineage development is recorded. Finally, Pretorius et al. utilize cell lineages for analyzing cancer growth [58], which makes their work the most similar to ours. Their work uses lineages as a way to characterize differences in ongoing cell development and combines the tree data of the lineages with the temporal data in a way similar to ours. However, Pretorius et al. visualize event data instead of time-series data and do not incorporate images into the visualization.

[1] Since these visualizations build on the Loon ecosystem, we have selected another animal to represent them. The selection of aardvarks has no particular significance except that the authors like them.

2.2 Tree Visualizations

The use of trees in visualizations is a well-studied and active area of research [22, 23, 63, 64]. In addition to the various methods for visualizing trees, work has been done to combine network and tree visualizations with other dimensions of data [54]. These approaches share some similarities with Aardvark, but all differ in key ways.

First, Beham et al. incorporate images of generated geometries into a radial tree visualization [3]. However, the nodes of the tree lack time-series data, and although there are some nontemporal attributes, they are not included directly in the tree visualization.

Visualizations that combine trees with other attributes are more common in the literature. Nobre et al. visualize genealogy data with related attributes by aligning nodes of the tree with rows in a table [52]. A similar approach is taken in Juniper [55] for a more general-purpose tree plus attribute visualization. Dendrograms linked with heatmaps [20, 40, 67] also associate trees with attributes. In contrast to the trees we consider, dendrograms augment the heatmap by indicating the similarity between rows of attribute data. Phylogenetic trees are also sometimes visualized alongside a heatmap [6, 32, 36]. Here, the tree displays the relationships of species to compare them with the measured attributes. However, for both dendrograms and phylogenetic trees, no time-series or imaging data is visualized.

Some work has combined temporal data with trees. For instance, several approaches visualize changes in tree structure [10, 22, 31, 69]. However, we are interested in a single tree structure where attributes on each node vary over time. Icicle plots [34] are a classic tree visualization technique that is often used for function call profiling, but the node's temporal relationship differs from ours. Shreck et al. organize time-series charts in a space-efficient TreeMap layout [62]. This layout may be appropriate for hierarchically clustering similar time-series data. However, our work requires a more direct encoding of parent-child relationships. A few techniques are more similar to our own. Burch et al. align a tree with time-series data [8, 9] Similarly, Nobre et al. also show time-series data in a tree layout [53]. Although both of these approaches resemble our approach, neither incorporates imaging data.

2.3 Image Snippets

We use the term *image snippet* or *snippet* to refer to a region of an image or video that is cut or *snipped* from its source and displayed in other contexts. Snippets are useful for extracting interesting parts of an image or video, especially if the original image is large or complex.

Small but important regions can be identified and extracted from high-resolution images. Lekschas et al. apply this principle to high-resolution images [38] and genome interaction matrices [37]. Ghani et al. do the same with network visualizations [21]. The image datasets we work with are not high-resolution; however, extracting individual cells into snippets enables combining them with other data types.

Complex multichannel images may have too much information to display simultaneously, requiring an interactive approach. Jessup et al. use the concept of a scope to interactively display a snippet at a higher resolution with different channels highlighted [27]. In contrast, our approach focuses on combining images with their derived metadata, not on understanding multiple image channels at once.

When interacting with data to either correct errors or modify designs, snippets can serve as a preview for possible user actions. Choi et al. show snippets of classification recommendations for microscopy images [14]. Similarly, Coffey et al. use preview snippets to show potential design alternatives for medical devices [16]. However, our approach is focused on the display of data currently present and does not incorporate modifications to the data.

When there are many images, exemplars can be used to represent an entire group. Lekschas et al. group similar images into *piles* and show an exemplar image on top to represent that pile [39]. In Loon [35],

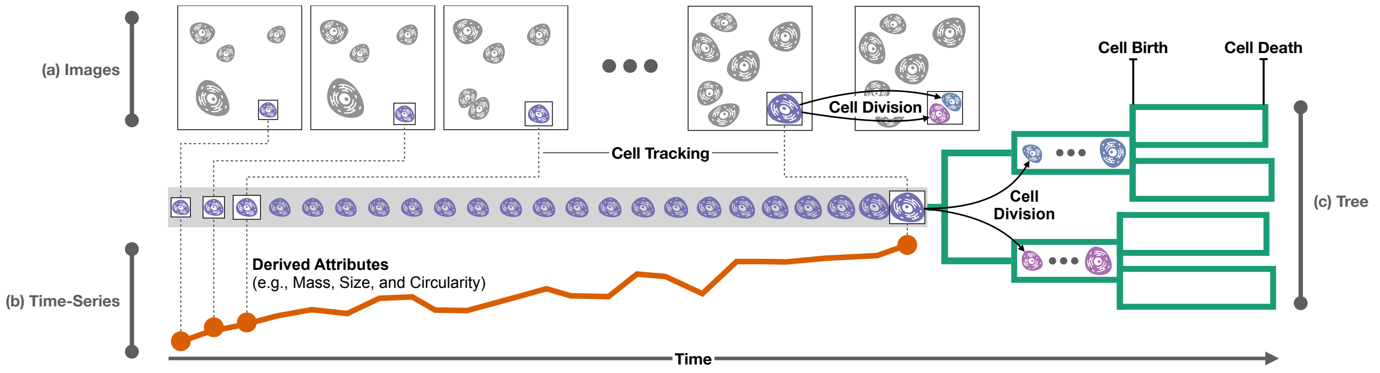

Fig. 2: Illustration of the data acquisition pipeline. (a) Images of cells (violet) over time are the input data type for the pipeline. (b) Cell segmentation produces outlines of cells and various derived attributes (orange), such as the area, mass, or shape of the cell. (c) Lineages (family trees, green) of cells are constructed by observing cell divisions and matching the daughter cells with the parent cell.

exemplar cells are selected for each experimental condition by sampling from metadata distributions. In our work, we focus on designing a fundamental detailed visualization for a single lineage.

For long videos, select frames can produce a static array of images that summarize the most important developments of the video. Lue et al. use this idea to summarize news videos [41]. Similarly, Yang et al. identify key frames in surveillance videos [75]. In our work, we identify key frames in a cell's development to summarize it.

3 Data

In this section, we describe the data Aardvark was designed to visualize. We provide essential details on how data is collected and processed in a time-series microscopy experiment (Figure 2).

Our collaborators take several steps to collect and process cell lineage data. First, **Images** of cells are taken repeatedly over the course of hours to days (Figure 2a). Next, cells are segmented to find their boundaries, resulting in *segmentations*. Cell segmentations are then tracked over time, resulting in *cell tracks* that ideally capture the lifetime of a cell. Attributes, such as mass, area, shape descriptors, fluorescence, and spatial positions, are derived from these segmentations. These derived attributes over time and cell movement comprise the **Time-Series** data (Figure 2b). Finally, when a parent cell divides into two daughter cells, that division is identified, and the relationship between the cell tracks is recorded. This **Tree** of cell relationships includes the start and end time at each node. We refer to a tree as a *cell lineage* (Figure 2c).

We illustrate our work with three datasets that utilize cell lineages to inquire about fundamental scientific questions. Each dataset contains 2,000–100,000 cells, 30–4,500 cell tracks, and 4–156 lineages. The largest lineages for each dataset are tracked 5–13 generations and contain 9–219 cell tracks. The imaging data contains 200–3,000 imaging frames at a resolution between 160x160 and 800x800. The total size of the datasets ranges between 380 MB and 1.25 GB. We will describe the individual datasets in more detail in Section 8.

4 Domain Tasks

This work results from a four-year and ongoing collaboration with the Zangle Lab and the Judson-Torres Lab [35]. The first author attended group meetings and worked closely with graduate students and postdocs in these labs, formatting data and debugging code. Furthermore, while designing and implementing Aardvark, the authors met on a recurring basis to discuss ideas and feedback on the tool. A static version of this visualization has already been utilized in domain-specific work submitted for publication [77] by our collaborators. Throughout this collaboration, we identified the following domain goals.

4.1 Quality Control Tasks

Ensuring data quality is a difficult but crucial task. We identified four subtasks that are useful to discuss separately but are inherently intertwined, as issues with one aspect of the data tend to affect others.

QC-Segmentation Although cell segmentation techniques are improving, it is an ongoing challenge largely because cancer cells are highly heterogeneous — they come in many shapes and sizes, which makes a one-size-fits-all algorithm difficult to construct. Cells can also overlap, and debris can influence tracking. Therefore, domain experts must review segmentations to ensure that the current algorithm performs within expectations.

QC-Attributes Once cells are segmented, attributes of the cells can be derived. The types of attributes that can be measured depend on the particular instrument. Common attributes include mass, fluorescence, or a measure of roundness. These attributes are used, for instance, to see if different drugs affect cells' change in mass. Ensuring that the attributes are calculated correctly is essential for sound conclusions based on these analyses.

QC-Tracking To measure cell changes over time, a segmented cell has to be tracked across frames, which can be challenging since cells can move out of frame or divide. Common issues include recording two different cells as one incorrect *merged track* or temporarily losing track of a cell, resulting in *broken tracks*.

QC-Lineage To study how cells proliferate, it is necessary to track cell lineages by identifying cell divisions and recording the correspondence between the parent and daughter cells. All the problems with segmentation and tracking can influence lineage tracing, but there are additional hurdles. For example, an algorithm can miss a cell division or record an incorrect division.

4.2 Discovery Tasks

Our collaborators want to study cell growth and propagation and the factors that influence it. In the following section, we list representative domain tasks they are interested in.

D-Propagation When cells divide, they do not always split their cell material evenly between the two daughter cells. When this happens, how does it affect later generations? For instance, if one cell receives a small amount of a particular protein, will later generations continue to be deficient in this protein, or will they bounce back and return to normal levels?

D-Cell Cycle Cells go through "phases" in their lifetime, referred to as the cell cycle. For example, cells get rounder and more compact shortly before division. However, there are also cell attributes where the pattern before division is unknown and of interest.

D-Changes Generally, cells grow and change gradually. Thus, deviations from this behavior, such as drastic changes in cell morphology, are of interest to our collaborators. For instance, a large drop in mass can indicate that a cell is dying.

D-Comparison Identifying differences between cells is a common goal. For instance, analysts might ask how cancer cells respond to different drugs or how different branches of a cell lineage develop differently to the same drug. To answer this question, domain scientists may want to identify divergences in any of the available data types across conditions, even between branches of a tree.

D-Synchrony Healthy cells generally have roughly the same lifespan: two daughter cells born at the same time should undergo phases of the cell cycle synchronously. In contrast, unhealthy cells or cells exposed to drugs may exhibit more heterogeneity. Therefore, determining the synchrony of cells is of interest.

4.3 Communication Tasks

Our collaborators need to share their findings with their peers. Communicating findings of complex data, as described in this paper, can be as challenging as quality control and discovery tasks. We designed Aardvark with communication in mind and list specific tasks below.

COM-Explain Scientific work often involves complex experiments, ideas, and data. In particular, the data for cell microscopy includes multiple data types that each tell one part of the story. Distilling these data types into figures for use in papers and presentations is critical to conveying the main concepts.

COM-Trust Understanding the message that authors are communicating is not enough. Readers must also have enough information to make a judgment about whether they trust the findings. Here, trust means that the audience can see evidence that supports their belief that the findings in the work are correct. We focus only on trust in the data, not the myriad of other factors that affect trust.

4.4 Task Abstraction

These domain goals can be mapped to abstract visualization tasks. The most fundamental task for any of these is the **synthesis** of the three data types: Relating data elements between the three types and visualizing them together is the primary visualization challenge we address in this paper. In combination with this synthesis, there are three tasks: analyzing the **topological** structure of the cell lineage, viewing the **trends** of the cell attributes, and understanding the **spatial relationships** of cells. These abstract tasks can address the different domain goals. For instance, errors in the lineage data can be identified by examining the *topology* and cross-checking it with the cells *spatial relationships*. Similarly, understanding how cell attributes change over generations requires viewing *trends* in those attributes and investigating how those trends relate to the *topology*. Consequently, these visualization tasks are the primary drivers for our design principles (Section 5) and Aardvark designs (Section 6). We illustrate how these choices tie back to the lower-level tasks in the case studies (Section 8)

5 Design Principles for Visualizing Trees, Time-Series, and Images

In this section, we describe design principles for integrating three types of cell microscopy data (trees, time-series, and images). Understanding the full picture of cell development requires analyzing all these data types together. Consequently, our visualization designs must also represent the different data types together. In our design process, we adopt Javed and Elmqvist's design space for composite visualizations. According to them, composite visualizations "combine multiple visualizations in the same visual space" [26]. We use this design space and describe considerations about how to choose visual encodings. We then introduce three designs (tree-first, time-series-first, and image-first) that implement these design principles for our data and tasks.

As part of our design process, we **first identify a primary data type** and select a visual encoding suitable for that data type. This primary encoding serves as the *host* visualization. **Next, the secondary data types are embedded into this chart as *client* visualizations**, and **visual encodings are selected.** We use a combination of *superimposed views* and *nested views* (Figure 1a) for our client visualizations, depending on the affordances of the host view. Superimposed views place the client visualization within the same space as the host view. Nested views nest the client visualization within the marks of the host visualization. In addition, when it is necessary to minimize overplotting in the host visualization, we show detailed charts on interaction, which we consider *superimposed detail on demand views*.

The choice of primary data type for a visualization depends on the analysis task. Although all three types are required for most analyses, some questions still prioritize certain features of the data. For instance, an understanding of the cell lineage topology will benefit from an explicit rendering of a tree diagram showing cell division. Alternatively, comparing the growth curves of multiple cells benefits from showing them in a line chart. Finally, spatial relationships of cells and their movements are best shown in a view that prioritizes the images of cells.

In practice, the choice of visual encoding and composition is not strictly sequential but **iterative and interleaved**. For example, when choosing a visual encoding for the host visualization, we ask the question of whether the encoding can accommodate the desired composite views. The layout and appearance of a node-link tree visualization, for example, can be modified to make space for nested or superimposed composite views without occlusion. Similarly, the choice of a visual encoding may limit the options available for client visualizations. For example, if the primary data type and host visualization is a space-filling image, any composition will lead to overlap. In this case, small and unobtrusive client visualizations or on-demand superimposition are good choices. Finally, host visual encodings should be chosen to work within the constraints of the clients, which typically have reduced space. For example, it might be prudent to choose compact visualizations that are still useful if just a few pixels are available.

Since our collaborators are interested in all these questions and more, we designed three composite visualizations, each encoding a different primary data type. Furthermore, each visualization can be *juxtaposed* and linked. The rest of this section discusses the high-level design decisions for each of these composite views, and the next section gives details for their instantiation in Aardvark.

5.1 Tree-First Visualization Design

The tree-first visualization (see Figure 3) uses the tree data type as the primary encoding. The nodes of the tree represent not only topology but also the lifetime of cells. Time-series charts are then nested inside the nodes of the tree diagram. Finally, images are superimposed along the time-series chart as space allows, and user-selected images are shown on demand. The composition of this view is illustrated in Figure 1(b).

We use an explicit node-link representation for our *tree data*. The horizontal alignment and width of nodes are determined by the start and end times of cell tracks. Assuming correct data, these times correspond to the cell's birth and division or death. The height of the node and the space between nodes can be adjusted based on analysis needs.

Nesting Time-Series. Within the node, we nest charts visualizing time-series data. As nodes are positioned and sized according to their birth and death time, the embedding and comparability of time-series data is straightforward. Various choices are available to encode the time-series data, such as line charts or color maps, but we use horizon charts [60] to encode the data. Horizon charts scale to vertically compact spaces, which is important for visualizing large trees and showing multiple time-series simultaneously (Figure 10). The

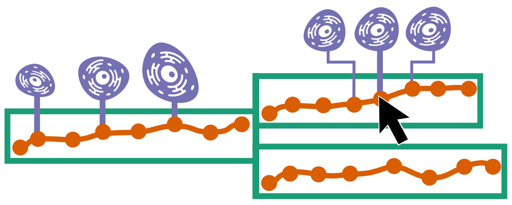

Fig. 3: Schematic of the tree-first visualization design. The primary data is the lineage, i.e., the tree capturing the relationship between the parent and the daughter cells (green). The horizontal sizes of the nodes are scaled to correspond to the cells' lifetime. A time-series dataset (orange) is nested, and cell images (violet) are superimposed, either using an automatic selection algorithm (left) or on demand (right).

redundant encoding of discrete bands using color further increases scalability for cases when the chart is only a few pixels high, at which point color is the dominant visual encoding. A second advantage of using redundant color in the tree is that color-coding along tree nodes is often familiar to domain scientists, unlike horizon charts.

Superimposed Images. We include images of individual cells superimposed along the horizon charts to provide context for the time-series and topology data. Cell images are clipped from the *image data* based on segmentation information. Since the images are a secondary data type in this view, their size and position are constrained by the host visualization. Thus, it is generally impossible to show every cell, and a selection strategy to choose exemplars must be employed [35]. We considered various sampling methods, including sampling frames evenly throughout the cell's lifetime. This approach is useful if cells change gradually throughout their life. However, regular sampling may miss a critical development that occurs in the span of a few frames. Therefore, we developed an alternative approach to select exemplars based on importance metrics computed from the other two data types. Different metrics are conceivable, but we employ two for our use case. The first is based on highlighting topologically relevant frames (**D-Cell Cycle**): we include frames right before or after cell division (at the beginning and end of the node). The second metric is based on changes of features of cell attributes, so that cells undergoing rapid change are included (**D-Changes**), or errors can be identified (**QC-Lineage**) as seen in Figure 4.

Superimposed Images on Demand. Although a data-driven selection strategy for exemplars provides a good starting point for showing relevant cells, some scenarios benefit from showing additional cells on the fly. Hence, we provide two methods to show superimposed cells on demand. First, free-form selection along the timeline enables an analyst to show any frame for a cell. Second, we provide keyframes on demand before and after selected cells. The latter is useful to observe the different stages of the cell cycle around an important point, such as a division. If the main frame shows a cell in the process of division, the prior frame will show the cell just before it divides, and the subsequent frame will show the newly divided cells.

5.2 Time-Series-First Visualization Design

The time-series-first visualization prioritizes the derived cell attributes. Attributes of interest commonly vary between studies (e.g., mass, fluorescence, shape), yet the analysis tasks for these time-series are similar. The most fundamental task is understanding the change of an attribute for a single cell over time. For instance, an analyst may ask, *is a cell growing or shrinking?* Next, the analyst may want to understand the behavior of a group of cells. *Are all cells growing, shrinking, or is there diverging behavior?* Line charts are well suited for this detailed comparison of time-series data and thus serve as our host visualization (see Figure 5). The questions about cell attributes usually extend across generations. For example, analysts might ask whether a parent's characteristics are predictive of the daughter cells' attributes (**D-Propagation**), e.g., *For cells that grow quickly, do their progeny also grow quickly?*

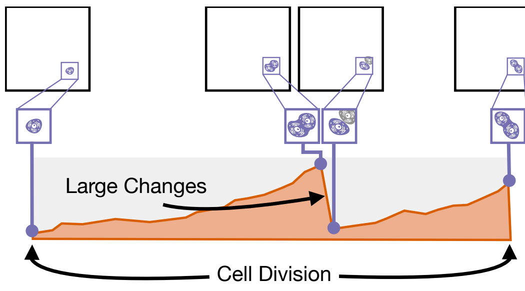

Fig. 4: Illustration of cell snippet selection. Cells snippets are extracted and shown at the beginning and the end (far left and far right) of the life of a cell. Additional snippets are shown when a large change in an attribute is detected. In this example, the attribute experiences a sudden drop. The associated cell image indicates that the reason is a cell division that was missed by the algorithm.

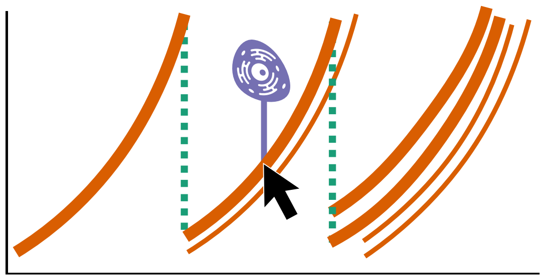

Fig. 5: Time-series-first visualization design showing attributes of individual cells as line charts (orange). As cells divide, the original cells' line ends, and new lines representing the daughter cells begin. Topological information about lineages is shown using superimposed on-demand composition for selected cells (green): the selected cell is connected to its parent with a dashed line. Lines corresponding to ancestors and descendants of a selected cell are also shown in bold. Cell images are shown using superimposed on-demand composition by rendering a cell image at a selected time-point (violet).

Superimposed Tree on Demand. A node in the line chart is represented by a continuous line. Hence, the node positions of the tree are determined by the attributes. As a consequence, plotting connections between nodes can result in visual clutter. Therefore, we do not attempt to show full topological information in the time-series-first view but instead show ancestry and descendants through superimposition on demand by connecting the end of one cell's line with the start of their daughter cell's line in the chart (see Figure 5 in green). To emphasize this relationship, we also display ancestors and descendants of a selected cell in bold.

Superimposed Images on Demand. Since line positions are driven by the data they represent, they cannot be arbitrarily repositioned. This constraint leads to challenges embedding images within the chart without obscuring the primary data type. Therefore, we chose to show cell images only when users select a specific feature in the data and then superimpose the image on demand in the form of a tool-tip visualization (see Figure 5 in violet).

5.3 Image-First Visualization Design

The image-first visualization, illustrated in Figure 6, uses the image data as the primary encoding. Images are especially useful for reviewing the spatial relationships of multiple cells and how cells move through space. For experiments that track cell lineages, combining all three data types is necessary to understand the relationship between cell movement and cell divisions. As an image view is a space-filling visualization, adding composite views will lead to occlusions. Hence, we choose to superimpose attributes that fit into the same coordinate space over embedding nested visualization thumbnails.

Superimposed Time-Series. Cell movement can be reduced to a trajectory of the cells' center of mass, resulting in a time-series of locations. Since this time-series data shares the same spatial coordinates as the image data, the trajectories can be superimposed on the image. This approach of superimposition hence follows the "eyes over memory" guideline of visualization design [50] of explicitly showing a temporal relationship in a static image, over-relying on a temporal animation, where viewers would have to memorize prior locations.

Superimposed Tree. Cells in an image commonly are closely related, and understanding that relationship is important for many analysis questions. To show these relationships directly in the image data, we superimpose the lineage tree on the images (see green marks in Figure 6). We use a node-link representation where the node

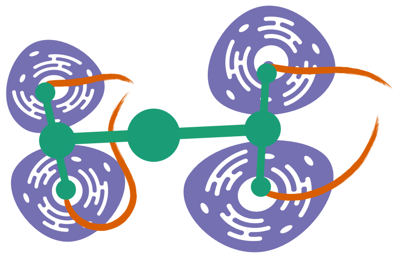

Fig. 6: Image-first visualization showing a full image with multiple cells (violet). Time-series data, in the form of cell location over time (orange), is superimposed, enabling analysts to understand movement over time in a static image. Lineage trees are also superimposed (green), showing relationships between cells.

positions are determined by the cells. Note that only the "leaves" of the tree can exist in an image; the parent cells have divided and do not exist anymore and are instead represented as circular marks. We also considered alternative layouts, such as implicit tree layouts [64], but found that the node-link representation best shows the relevant topology. Note that the tree uses a partially fixed layout, since the positions of the leaves are given by the cell location (parent cells can be freely placed). This restriction will make the topology less evident in most cases, illustrating the tradeoffs present when embedding these secondary visualizations in their client view.

5.4 Juxtaposition

Each of these composite visualizations has its strengths and weaknesses. Juxtaposing all three views together creates a system that can tackle a wide range of analyses. Linked and brushing ties the data elements together across these views. Since each view shows every data type, there are many opportunities to link data elements together. For instance, if a secondary data type is superimposed on demand in multiple views, triggering one view to show the superimposed chart should trigger all views. In our designs, images are superimposed on demand in the tree-first view and the time-series-first view. Thus, when a snippet is shown in the tree view, the same snippet is shown in the time-series view. However, the same principle for linking applies even if the data types, visualization encodings, or composition techniques differ.

6 Aardvark Design

In the previous section, we described our design process and the high-level design decisions of choosing visual encodings and view compositions to integrate the different data types. These designs are intended to be transferable to, and instructive for, other similar scenarios. In this section, we provide details on how we instantiated these designs in our prototype, Aardvark, and provide sufficient details to ensure the reproducibility of our implementation. Figure 9 shows examples of how these designs are implemented in the tool.

6.1 Tree-First Visualization Details

Tree Layout. Several user-configurable options determine the tree layout, allowing analysts to balance priorities between data types. A generously spaced layout will ensure enough room above each horizon chart to display the images for that cell. Alternatively, a dense layout will produce a compact view of the tree structure but will display fewer image snippets. Similarly, adjusting the height of horizon charts is a tradeoff between a larger tree with more detailed time-series data and a more compact tree showing less detail. The choice between a detailed and dense view is not binary but rather a spectrum that balances priorities between the three data types.

Cell Snippet Extraction. Cells are extracted from the source image by centering the cell and copying a constant number of pixels around the center. The number of pixels is the same for all cells so that the display size and scaling are consistent across snippets. One side effect of this approach is that the cell boundary may not fit completely

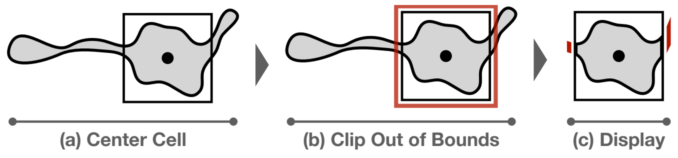

Fig. 7: Clipping of cells with large peripheral features. Some types of cells have features that extend far beyond the cell core. Rendering the whole cell within a small embedded view would result in barely visible features. To address this problem, we (a) first determine the center of the cell, (b) then clip the features outside of the bounds of the core, and (c) display the clipped cell with indicators that features have been clipped (red).

inside the region that is extracted, which frequently happens if cells have elongated shapes. In this case, we indicate where the cell is cut off with red lines (Figure 7). These snippets are then placed above or below the horizon chart, depending on available space.

Interaction. Aardvark supports superimposed charts on demand in a few ways. Hovering on the horizon chart will show the snippet at that time point, and the cell boundary clipping is disabled, so the full segmentation outline can be seen. Selecting a time point will pin the snippet in place. Hovering on existing snippets will show the previous and next snippets, even beyond the cell track (Figure 9f).

6.2 Time-Series-First Visualization Details

The line chart view within Aardvark supports different modes of aggregation — from visualizing individual cell attributes to population-wide aggregation. In the time-series-first visualization (Figure 9b), time is always mapped to the x-axis, and attributes are mapped to the y-axis. Selecting a line will show a snippet of that cell above the line (Figure 9d). If a line is selected, lineage information is displayed for cells with direct relationships to the selected cell through explicit rendering with dashed lines or emphasis and color highlighting, as described in Section 5.2. To help distinguish between different branches of progeny, the two subtrees of the selected cell are assigned a different color.

6.3 Image-First Visualization Details

Aardvark supports four imaging layers. The base microscopy **images** record pixel intensities that can have different meanings (mass, fluorencense, etc). A colormap is applied with an adjustable range so the signal in the data can be highlighted while reducing noise. **Cell boundaries** are displayed with an outline. **Cell trajectories** are shown as a line that fades into the past, as shown in Figure 9h. **Cell lineages** are displayed with a node-link diagram where internal nodes represent cell ancestors. Figure 8 illustrates this process across four generations.

6.4 Across View Interaction

There are many interactions across views in Aardvark, most importantly based on selecting a cell. Selecting a cell also selects a cell track, selects a time or image frame, and selects a cell lineage. In Aardvark, each view has special logic for how to display a selected cell, track, lineage, and time. All these selections are highlighted. In the *tree-first visualization*, the selected cell snippet is displayed (Figure 9d). All cell

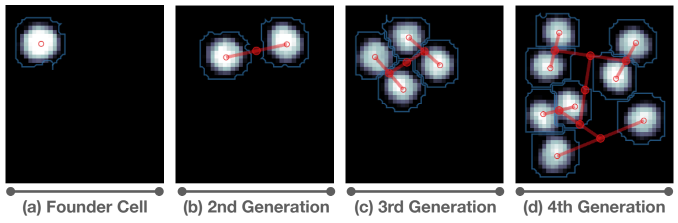

Fig. 8: Image views illustrating cell division across four generations and the overlaid cell lineages.

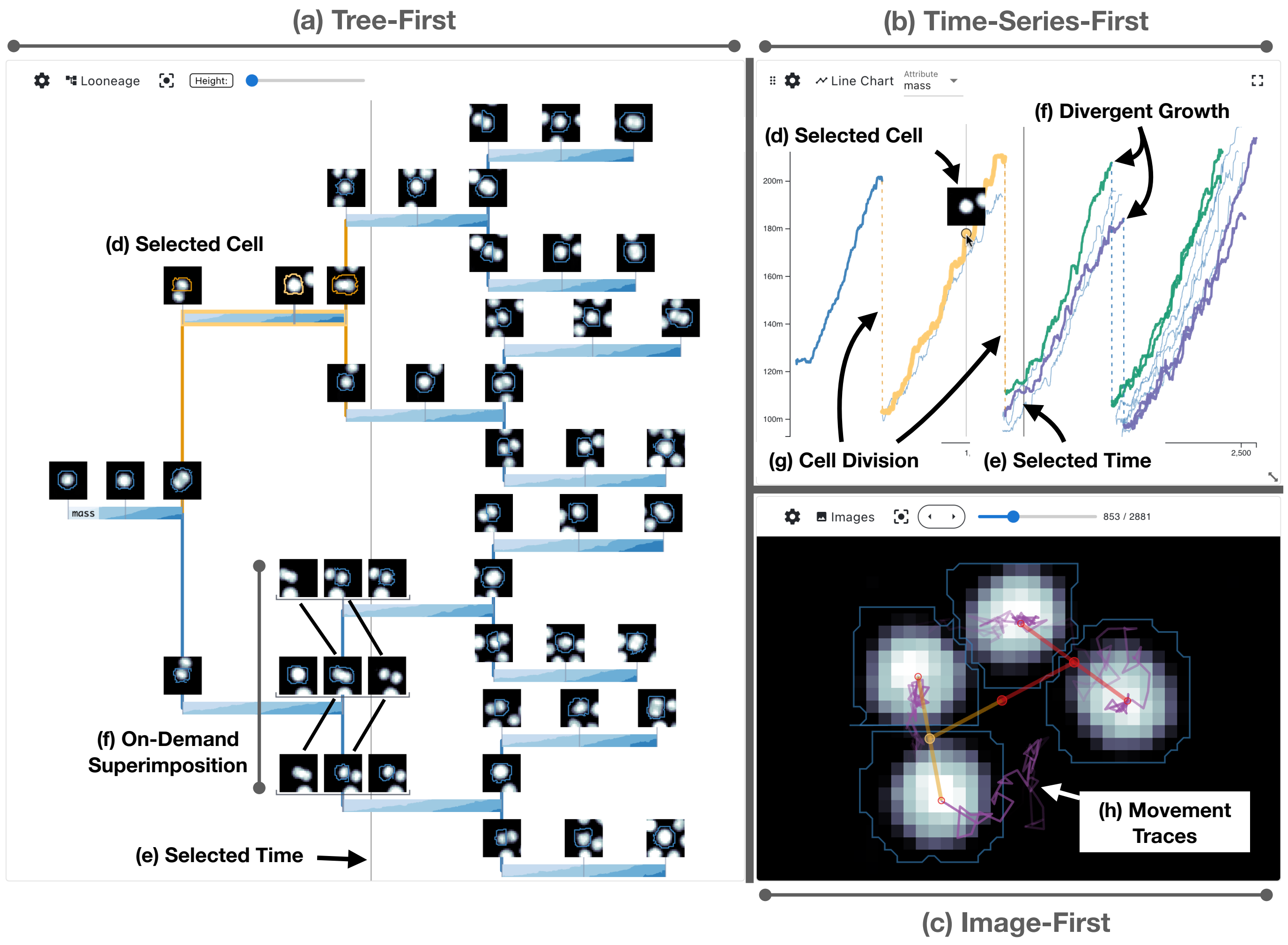

Fig. 9: The three visualizations as implemented in Aardvark. (a) The tree-first view shows cell growth and cell image snippets. The node at level two at the top is highlighted in orange. (b) The time-series-first view shows the highlighted cell in orange. The daughters of the selected cell show different growth behavior as evident from both the line chart and the horizon charts. (c) The image-first view shows the four leaf cells in the tree. The exact frame is highlighted by a vertical line in both the tree and time-series view. The lineage and the spatial movement are also shown.

snippets in the selected track have their cell boundary highlighted. The horizon chart of the selected track and its connecting lines are highlighted. Finally, a vertical line indicates the selected time (Figure 9e). In the *image-first visualization*, the cell boundary is highlighted if the selected cell is in the current frame. Otherwise, if the selected cell's progeny are in the current frame, the node and connecting edges in the tree representation are highlighted (Figure 9c). In the *time-series-first visualization*, the selected cell is shown as a snippet (Figure 9d). The line corresponding to the selected cell track is highlighted, as are the edges directly connected to the selected cell. Finally, the selected time is indicated with a vertical line (Figure 9e).

6.5 Other Views

Aardvark includes other useful views. **Tables** provide access to raw values for cells, tracks, and lineages. Sorting data columns provides a means to select lineages with specific characteristics. Displaying **dataset metadata** provides basic context and sanity checking for a dataset. Finally, the **visualization interaction state** is tracked and displayed for state recovery and provenance [17].

7 Implementation

Aardvark is implemented as an open-source front-end application available with demo datasets at `https://aardvark.sci.utah.edu/`. Imaging and metadata are fetched from a file server that can be specified to support various setups. For example, the demo datasets associated with this paper are stored on an AWS S3 bucket, but the tool can also be configured to access files stored locally.

Aardvark uses various libraries and web technologies: TypeScript, Vue 3, Pinia, and Quasar comprise the base framework and UI library; deck.gl serves as the base WebGL framework for the image and lineage charts. Components from Viv are used to load and render standard microscopy image formats as layers in deck.gl [44]. These are combined with custom deck.gl layers developed for this project.

We use utility functions from the D3 library [5] and the D3 Flextree plugin, which extends the tree layout module of D3 to produce a compact layout with variable width nodes [72]. Finally, the Trrack library is used to record and display interaction provenance [17]. For a complete list of libraries and the full source code, refer to `https://github.com/visdesignlab/aardvark`.

8 Case Studies

Aardvark was designed with our long-term collaborators (who are also co-authors), who have real data and real scientific questions [66] related to cancer cell development. The following case studies are selected examples collected over the course of the collaboration intended to illustrate the utility of our design with real data.

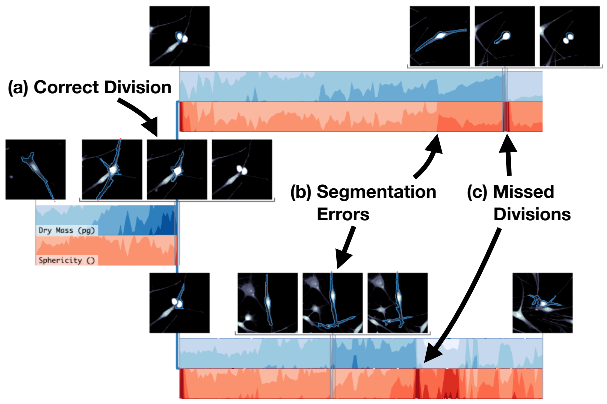

Fig. 10: Shows on example of (a) correct division, (b) multiple segmentation errors resulting in rapid changes in the attributes, and (c) two missed divisions. Immediately before the missed division, mass and sphericity increase, but then, one of the children is incorrectly tracked as its parent.

8.1 Quality Control: Cancer Triggering Microenvironments

This dataset is part of a study that examines the difference between the development of benign moles and melanoma [46]. Understanding the steps that initiate malignant disease could reveal potential chemopreventative strategies for skin cancer. The study exposed healthy human skin cells to a chemical that simulates the environment skin cells experience when exposed to the sun (UV radiation).

As a first step, we browse lineages that exhibit a large drop in mass and show it in the tree-first view. Since our image selection prioritizes data points with large changes (**D-Changes**), image snippets at points of change are automatically shown in the tree-first view, which quickly reveals why the drop in mass occurs. Some instances occur because the cell moves out of the imaging view (**QC-Tracking**). Some occur because part or all of a nearby cell is incorrectly included in the cell (**QC-Segmentation**). Some occur because a cell divides, but tracking is incorrectly connected to one of the daughter cells (**QC-Lineage**,**QC-Tracking**). Understanding which of these errors occurs requires a combination of all the different data types. The tree-first view is well suited for quickly making these identifications. In Figure 10, several errors can be seen in one lineage. This figure shows two cell attributes in the horizon charts (mass and sphericity), which are informative for quality control because their behavior within the cell cycle is well understood (**QC-Attributes**). The first cell division recorded in this example appears correct (Figure 10a). The image segmentations are in alignment (**QC-Segmentation**), and the cell attributes show the expected response — cell mass increases throughout the cell's life, and cells are spherical just before and after division (**D-Cell Cycle**). In the second generation, there are similar patterns in the attributes, but closer inspection reveals different reasons and types of errors. Figure 10b shows a sudden change in mass. Inspection of the cell images reveals that this is a segmentation artifact from a cell "limb" being excluded and included in different frames. Figure 10c highlights two missed divisions. In both cases, mass and sphericity increase and then suddenly drop. Inspecting three frames at this point reveals that the cell divided, and one of the daughter cells is tracked incorrectly as its parent cell.

QC issues can also be identified in the image view. Segmentation errors can be spotted by matching the segmentation outline to the image. Divisions can be validated using the superimposed tree and location data. Figure 11a shows a clipped example of a correctly identified division. The two cells are connected, indicating that they share a parent, and the location traces show that they both came from the same origin. Figure 11b shows an example with similar image and location trace data, but the tree connection is missing. Navigating to an earlier frame reveals that these cells are from a common parent.

These examples illustrate the variety of issues that can be quickly identified in Aardvark as a first step toward addressing them.

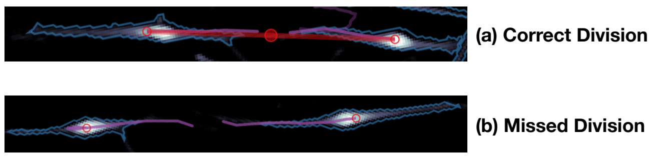

Fig. 11: Reviewing tracking of cell divisions in the image view. (a) A correctly tracked division can be identified by the visible tree indicating that the cells are siblings and the location traces showing that they originated at the same place spatially. (b) The lack of a tree indicates that no division was recorded, yet the location tracks show that they did indeed originate at the same place.

8.2 Data Discovery: Automated Cell Lineage Tracking

Producing cell lineage data is challenging. Automated systems can be error-prone, and manual reconstruction can be time-intensive. This dataset is from work that demonstrates a technique for producing high-quality lineages automatically [76]. In short, this technique separates individual cells into small microwells where they can grow and divide in isolation. These microwells ensure that cells remain within the microscope's view and that all cells within a microwell originate from a single founder cell. The data shown in this paper is focused on a single microwell containing a mouse leukemia cell lineage.

Before interpreting the data collection and processing results, a critical first step is to verify that the lineage data is correct. For this dataset, lineage tracking is expected to eventually fail when the number of cells in the microwell leads to cells overlapping in 3D, which makes them impossible to reliably separate. A question our collaborators are asking is, hence, how many generations can be accurately tracked. We can approach this question in Aardvark by iteratively exploring the dataset for an individual lineage. The tree-first view provides an initial overview of the data (Figure 9a). The default view shows that every cell starts at roughly the same mass and has a similar growth rate, which is expected for this dataset (**QC-Attributes**). Image snippets of the cells are automatically shown for each division point. Selecting a snippet just before cell division reveals the next frame (Figure 9f), which makes it easy to verify that the three cells (parent and two daughter cells) are recorded correctly (**QC-Lineage**). Interactively expanding the width of the tree allows more space for additional snippets to be shown. These snippets can be quickly scanned to verify that the segmentation and tracking of the cell is consistent (**QC-Segmentation**,**QC-Tracking**).

After verifying the quality of our dataset, our collaborators start investigating biologically interesting patterns. They select a cell in the second generation (Figure 9d), which updates the time-series-first view to show part of the lineage tree (Figure 9b). The distinct coloring of the two branches reveals a difference between them (Figure 9f, **D-Propagation**). Our collaborators conclude that data such as these could be used to assess drivers of asymmetric division and heredity by, for example, using genetic mutants of key growth regulation pathways.

8.3 Communication: Tumorigenic Cell States

Our collaborators study how skin cancer cells (melanoma) leave the primary tumor and form new tumors in other organs (metastatic dissemination) [77]. Cells that can form new tumors are referred to as tumorigenic cells. However, cells are not predestined to be tumorigenic from their genetic material. Instead, the tumorigenic state of a melanoma cell can change within a cell's lifetime or across generations. This state can be measured by engineering the cells to express a fluorescent marker, mCherry, under the control of a specific promoter. Low levels of mCherry indicate that the cell is in a tumorigenic state. In this dataset, a combination of automated and manual processing was used to construct several lineages. These lineages illustrate how this tumorigenic state can change across generations.

Figure 12 shows an example of this effect: A distinct asymmetry in the tree topology is apparent in the tree-first view (compare the long lifetime of the cell in Figure 12b with the short lifetime of its sister cell and its descendants (Figure 12a). The cells in the top branch are

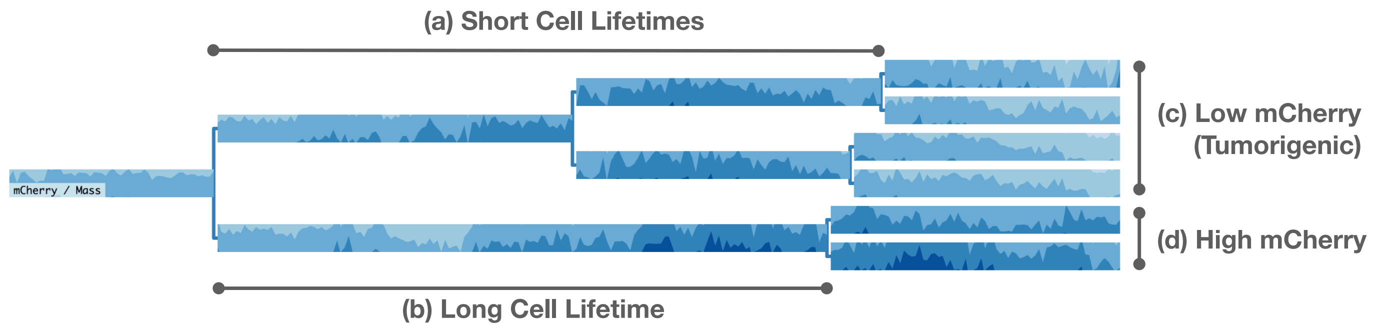

Fig. 12: Example of the emergence of tumorigenic melanoma cells in a single lineage. Notice the distinct asymmetry of the tree: cells in the top branch (a) live about half the time before they subdivide compared to the cells in the bottom branch (b) — cancerous cells tend to grow faster than benign cells. The embedded horizon charts show mCherry, a fluorescent marker of tumorigenesis normalized by cell mass. Low mCherry indicates a tumorigenic cell. We observe a distinct differentiation between the cells in (c) the top branch (low mCherry) and (d) the bottom branch (high mCherry).

subdividing in about half the time as those in the bottom branch (**D-Synchrony**, **D-Comparison**). Within the tree-first view, the normalized abundance of the fluorescent marker is shown, *mCherry / Mass*. This marker shows the relative amount of mCherry in the cell independent of its total mass, the latter of which varies throughout the cell lifetime (**D-Cell Cycle**). Since the horizon charts are nested within the tree structure, it can be seen that the later generations of the top branch end the experiment with a lower amount of *mCherry / mass* (Figure 12c). The short division time and low mCherry values indicate that this top branch of cells has transitioned into a tumorigenic state. This example illustrates how, within a single lineage, some, but not all, cells transition into a tumorigenic state. A similar figure from an earlier version of Aardvark is included in a preprint [77] (**COM-Explain**, **COM-Trust**), illustrating the value of Aardvark for scientific communication.

9 Discussion and Limitations

In this work, we have focused on visualizing data that combines trees, time-series, and images, where all the data is derived from the primary data type: time-series microscopy images. A major theme throughout this work is that understanding this data holistically requires understanding how these three pieces of information fit together.

Scalability. Our work has limitations, particularly when scaling to large trees with over a few hundred nodes. Although constructing trees of this size for cell lineage tracing is technically difficult, it is conceivable for similar datasets from other domains. In such cases, displaying the entire tree with all cell images may not be feasible. To mitigate this problem, we enable interaction to adjust tree size and spacing. We also show only one tree simultaneously. Showing multiple trees would be useful for comparing or exploring a collection of lineages. We leave dedicated approaches for multiple lineage comparisons to future work.

Generalizability. Our work is relevant to those interested in cell microscopy data, but other domains with similar data combinations could benefit from our approach. Satellite imaging of Earth can have derived attributes from the images or be associated with data collected on Earth in similar regions. A subdivision of regions could produce a tree relationship. Astronomy imaging tracks and analyzes the change of celestial bodies. Analyzing data from smart cars could combine imaging and sensor data. Finally, physics simulations that track attributes, positions, and shapes of objects under different conditions could also use elements of our designs. More broadly, although this combination of data types is specific, combining data types is not. In particular, deriving secondary data from an imaging modality is a widely employed approach across disciplines. In biology, imaging is often combined with other approaches to reveal the complex mechanisms that comprise organisms. Our work would be a useful reference for problems that have a similar combination of data types. Yet, we argue that our design principles could still be applied to other data type combinations. For example, the process of selecting primary and secondary data types, their encodings, and their composition could be leveraged for any combination of data types. This framework for reasoning about the design options helps navigate the complex space of these multimodal datasets.

Evaluation. We considered several strategies for evaluating Aardvark. Since it is a specialized tool with a limited number of expert users, a quantitative evaluation is difficult. A quantitative study with a broader audience could potentially be performed for certain isolated components of our system. However, evaluating the full system in this way lacks ecological validity. Alternatively, we considered evaluating the tool with our current users. Since they have codesigned the tool with us and are authors of this paper, a study is susceptible to demand characteristics [7], i.e., introducing bias to give positive scores or feedback. Therefore, we decided that, in order to ensure the rigor [48] of our design study, we report factually on different scenarios where the tool has provided utility for our collaborators.

10 Conclusion and Future Work

In conclusion, this work examines three distinct data types (trees, time-series, and images) that are interwoven to create a complex multimodal dataset. We describe our design principles for combining these complex, disparate data types into intuitive composite visualizations. We use these principles to implement an open-source visualization tool, Aardvark. We demonstrate the utility of Aardvark to perform quality control, data analysis, and communication tasks with three case studies.

We plan to continue our collaboration with multiple potential research directions in mind. Providing a command line interface to create a static version of the lineage diagrams would help our design reach a wider audience of users, as past experience has shown that scientists tend to prefer tools that neatly fit into their workflow. On the flip side, developing Aardvark into an integrated system that combines image analysis (segmentation, tracking) with visualizations would allow us to identify errors in the data and immediately fix them, potentially improving the parameterization of the image analysis algorithms.

Acknowledgments

This work was supported by NIH R01CA276653 and NIH 3P30CA042014-34S6. The authors wish to thank Jack Wilburn for technical help, Koushik Roy, members of the Visualization Design Lab, the Scientific Computing and Imaging Institute, and the Zangle and Judson-Torres labs for their feedback and advice.

References

[1] A. K. Ai-Awami, J. Beyer, D. Haehn, N. Kasthuri, J. W. Lichtman, H. Pfister, and M. Hadwiger. NeuroBlocks – Visual Tracking of Segmentation and Proofreading for Large Connectomics Projects. *IEEE Trans. Visual Comput. Graphics*, 22(1):738–746, 2016. doi: 10.1109/TVCG.2015.2467441 2

[2] G. Anselmi and M. Humphry. Livecyte: Creating a comprehensive cell profile. *Microscopie*, 30(2), 2019. doi: 10.4081/microscopie.2019.8592 2

[3] M. Beham, W. Herzner, M. E. Gröller, and J. Kehrer. Cupid: Cluster-Based Exploration of Geometry Generators with Parallel Coordinates and Radial Trees. *IEEE Trans. Visual Comput. Graphics*, 20(12):1693–1702, 2014. doi: 10.1109/TVCG.2014.2346626 2

[4] J. Beyer, J. Troidl, S. Boorboor, M. Hadwiger, A. Kaufman, and H. Pfister. A Survey of Visualization and Analysis in High-Resolution Connectomics. *Comput. Graphics Forum*, 41(3):573–607, 2022. doi: 10.1111/cgf.14574 2

[5] M. Bostock, V. Ogievetsky, and J. Heer. D^3 Data-Driven Documents. *IEEE Trans. Visual Comput. Graphics*, 17(12):2301–2309, 2011. doi: 10.1109/tvcg.2011.185 7

[6] B. R. Briggs, J. W. Pohlman, M. Torres, M. Riedel, E. L. Brodie, and F. S. Colwell. Macroscopic Biofilms in Fracture-Dominated Sediment That Anaerobically Oxidize Methane. *Applied and Environmental Microbiology*, 77(19):6780–6787, 2011. doi: 10.1128/AEM.00288-11 2

[7] B. Brown, S. Reeves, and S. Sherwood. Into the wild: Challenges and opportunities for field trial methods. In *Proc. SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pp. 1657–1666. ACM, New York, May 2011. doi: 10.1145/1978942.1979185 9

[8] M. Burch, F. Beck, and S. Diehl. Timeline trees: Visualizing sequences of transactions in information hierarchies. In *Proc. Working Conference on Advanced Visual Interfaces*, AVI, pp. 75–82. ACM, New York, 2008. doi: 10.1145/1385569.1385584 2

[9] M. Burch and D. Weiskopf. VISUALIZING DYNAMIC QUANTITATIVE DATA IN HIERARCHIES - TimeEdgeTrees: Attaching Dynamic Weights to Tree Edges. In *International Conference on Information Visualization Theory and Applications*, pp. 177–186, 2023. 2

[10] S. K. Card, B. Suh, B. A. Pendleton, J. Heer, and J. W. Bodnar. Time Tree: Exploring Time Changing Hierarchies. In *2006 IEEE Symposium On Visual Analytics Science And Technology*, pp. 3–10, 2006. doi: 10.1109/VAST.2006.261450 2

[11] A. E. Carpenter, T. R. Jones, M. R. Lamprecht, C. Clarke, I. H. Kang, O. Friman, D. A. Guertin, J. H. Chang, R. A. Lindquist, J. Moffat, P. Golland, and D. M. Sabatini. CellProfiler: Image analysis software for identifying and quantifying cell phenotypes. *Genome Biology*, 7(10):R100, 2006. doi: 10.1186/gb-2006-7-10-r100 2

[12] A. Cedilnik, J. Baumes, L. Ibanez, S. Megason, and B. Wylie. Integration of information and volume visualization for analysis of cell lineage and gene expression during embryogenesis. In *Visualization and Data Analysis 2008*, vol. 6809, pp. 193–203. SPIE, 2008. doi: 10.1117/12.768014 2

[13] F. Cheng, M. S. Keller, H. Qu, N. Gehlenborg, and Q. Wang. Polyphony: An Interactive Transfer Learning Framework for Single-Cell Data Analysis. *IEEE Trans. Visual Comput. Graphics*, 29(1):591–601, 2023. doi: 10.1109/TVCG.2022.3209408 2

[14] J. Choi, H.-J. Oh, H. Lee, S. Kim, S.-K. Kwon, and W.-K. Jeong. MitoVis: A Unified Visual Analytics System for End-to-End Neuronal Mitochondria Analysis. *IEEE Trans. Visual Comput. Graphics*, pp. 1–15, 2022. doi: 10.1109/TVCG.2022.3233548 2

[15] S. H. Choi, E. J. Ku, Y. A. Choi, and J. W. Oh. Grave-to-cradle: Human embryonic lineage tracing from the postmortem body. *Exp. Mol. Med.*, 55(1):13–21, 2023. doi: 10.1038/s12276-022-00912-y 2

[16] D. Coffey, Chi-Lun Lin, A. G. Erdman, and D. F. Keefe. Design by Dragging: An Interface for Creative Forward and Inverse Design with Simulation Ensembles. *IEEE Trans. Visual Comput. Graphics*, 19(12):2783–2791, 2013. doi: 10.1109/TVCG.2013.147 2

[17] Z. T. Cutler, K. Gadhave, and A. Lex. Trrack: A Library for Provenance Tracking in Web-Based Visualizations. In *IEEE Visualization Conference (VIS)*, pp. 116–120, 2020. doi: 10.1109/VIS47514.2020.00030 7

[18] W. De Leeuw, R. Van Liere, P. Verschure, A. Visser, E. Manders, and R. Van Drielf. Visualization of time dependent confocal microscopy data. In *Proceedings Visualization (VIS)*, pp. 473–476, 2000. doi: 10.1109/VISUAL.2000.885735 2

[19] K. Dinkla, H. Strobelt, B. Genest, S. Reiling, M. Borowsky, and H. Pfister. Screenit: Visual Analysis of Cellular Screens. *IEEE Trans. Visual Comput. Graphics*, 23(1):591–600, 2017. doi: 10/f92c98 2

[20] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *PNAS*, 95(25):14863–14868, 1998. doi: 10.1073/pnas.95.25.14863 2

[21] S. Ghani, N. H. Riche, and N. Elmqvist. Dynamic Insets for Context-Aware Graph Navigation. *Comput. Graphics Forum*, 30(3):861–870, 2011. doi: 10.1111/j.1467-8659.2011.01935.x 2

[22] C. Han, J. Jo, A. Li, B. Lee, O. Deussen, and Y. Wang. SizePairs: Achieving Stable and Balanced Temporal Treemaps using Hierarchical Size-based Pairing. *IEEE Trans. Visual Comput. Graphics*, 29(1):193–202, 2023. doi: 10.1109/TVCG.2022.3209450 2

[23] T. Harbig, M. W. Paz, and K. Nieselt. GO-Compass: Visual Navigation of Multiple Lists of GO terms. *Comput. Graphics Forum*, 42(3):271–281, 2023. doi: 10.1111/cgf.14829 2

[24] G. Hattab, V. Wiesmann, A. Becker, T. Munzner, and T. W. Nattkemper. A Novel Methodology for Characterizing Cell Subpopulations in Automated Time-lapse Microscopy. *Front. Bioeng. Biotechnol.*, 6, 2018. doi: 10.3389/fbioe.2018.00017 2

[25] J. Hong, A. Trubuil, and T. Isenberg. LineageD: An Interactive Visual System for Plant Cell Lineage Assignments based on Correctable Machine Learning. *Comput. Graphics Forum*, 41(3):195–207, 2022. doi: 10.1111/cgf.14533 2

[26] W. Javed and N. Elmqvist. Exploring the design space of composite visualization. In *Proc. IEEE Pacific Visualization Symposium*, pp. 1–8. IEEE, 2012. doi: 10.1109/PacificVis.2012.6183556 4

[27] J. Jessup, R. Krueger, S. Warchol, J. Hoffer, J. Muhlich, C. C. Ritch, G. Gaglia, S. Coy, Y.-A. Chen, J.-R. Lin, S. Santagata, P. K. Sorger, and H. Pfister. Scope2Screen: Focus+Context Techniques for Pathology Tumor Assessment in Multivariate Image Data. *IEEE Trans. Visual Comput. Graphics*, 28(1):259–269, 2022. doi: 10.1109/TVCG.2021.3114786 2

[28] T. R. Jones, I. H. Kang, D. B. Wheeler, R. A. Lindquist, A. Papallo, D. M. Sabatini, P. Golland, and A. E. Carpenter. CellProfiler Analyst: Data exploration and analysis software for complex image-based screens. *BMC Bioinformatics*, 9(1):482, 2008. doi: 10.1186/1471-2105-9-482 2

[29] M. S. Keller, I. Gold, C. McCallum, T. Manz, P. V. Kharchenko, and N. Gehlenborg. Vitessce: Integrative visualization of multimodal and spatially-resolved single-cell data. *OSF Preprints*, 2021. doi: 10.31219/osf.io/y8thv 2

[30] A. Kerren and F. Schreiber. Why Integrate InfoVis and SciVis?: An Example from Systems Biology. *IEEE Comput. Graphics Appl.*, 34(6):69–73, Nov. 2014. doi: 10.1109/MCG.2014.122 2

[31] W. Köpp and T. Weinkauf. Temporal Treemaps: Static Visualization of Evolving Trees. *IEEE Trans. Visual Comput. Graphics*, 25(1):534–543, 2019. doi: 10.1109/TVCG.2018.2865265 2

[32] Ł. Kreft, A. Botzki, F. Coppens, K. Vandepoele, and M. Van Bel. PhyD3: A phylogenetic tree viewer with extended phyloXML support for functional genomics data visualization. *Bioinformatics*, 33(18):2946–2947, 2017. doi: 10.1093/bioinformatics/btx324 2

[33] R. Krueger, J. Beyer, W.-D. Jang, N. W. Kim, A. Sokolov, P. K. Sorger, and H. Pfister. Facetto: Combining Unsupervised and Supervised Learning for Hierarchical Phenotype Analysis in Multi-Channel Image Data. *IEEE Trans. Visual Comput. Graphics*, 26(1):227–237, 2020. doi: 10.1109/TVCG.2019.2934547 2

[34] J. B. Kruskal and J. M. Landwehr. Icicle Plots: Better Displays for Hierarchical Clustering. *The American Statistician*, 37(2):162, 1983. doi: 10.2307/2685881 2

[35] D. Lange, E. Polanco, R. Judson-Torres, T. Zangle, and A. Lex. Loon: Using Exemplars to Visualize Large-Scale Microscopy Data. *IEEE Trans. Visual Comput. Graphics*, 28(1):548–258, 2022. doi: 10.1109/TVCG.2021.3114766 2, 3, 5

[36] B. Lee, L. Nachmanson, G. Robertson, J. M. Carlson, and D. Heckerman. PhyloDet: A scalable visualization tool for mapping multiple traits to large evolutionary trees. *Bioinformatics*, 25(19):2611–2612, 2009. doi: 10.1093/bioinformatics/btp454 2

[37] F. Lekschas, B. Bach, P. Kerpedjiev, N. Gehlenborg, and H. Pfister. HiPiler: Visual Exploration of Large Genome Interaction Matrices with Interactive Small Multiples. *IEEE Trans. Visual Comput. Graphics*, 24(1):522–531, 2018. doi: 10.1109/TVCG.2017.2745978 2

[38] F. Lekschas, M. Behrisch, B. Bach, P. Kerpedjiev, N. Gehlenborg, and H. Pfister. Pattern-Driven Navigation in 2D Multiscale Visualizations with Scalable Insets. *IEEE Trans. Visual Comput. Graphics*, 26(1):611–621, 2020. doi: 10.1109/TVCG.2019.2934555 2

[39] F. Lekschas, X. Zhou, W. Chen, N. Gehlenborg, B. Bach, and H. Pfister. A Generic Framework and Library for Exploration of Small Multiples through Interactive Piling. *IEEE Trans. Visual Comput. Graphics*, 27(2):358–368, 2021. doi: 10.1109/TVCG.2020.3028948 2

[40] A. Lex, M. Streit, C. Partl, K. Kashofer, and D. Schmalstieg. Comparative Analysis of Multidimensional, Quantitative Data. *IEEE Trans. Visual Comput. Graphics*, 16(6):1027–1035, 2010. doi: 10.1109/TVCG.2010.138 2

[41] H. Luo, J. Fan, J. Yang, W. Ribarsky, and S. Satoh. Exploring Large-Scale Video News via Interactive Visualization. In *2006 IEEE Symposium*

On Visual Analytics Science And Technology, pp. 75–82, 2006. doi: 10.1109/VAST.2006.261433 3

[42] S. L'Yi and N. Gehlenborg. Multi-View Design Patterns and Responsive Visualization for Genomics Data. *IEEE Trans. Visual Comput. Graphics*, 29(1):559–569, 2023. doi: 10.1109/TVCG.2022.3209398 2

[43] S. L'Yi, Q. Wang, F. Lekschas, and N. Gehlenborg. Gosling: A Grammar-based Toolkit for Scalable and Interactive Genomics Data Visualization. *IEEE Trans. Visual Comput. Graphics*, 28(1):140–150, 2022. doi: 10.1109/TVCG.2021.3114876 2

[44] T. Manz, I. Gold, N. H. Patterson, C. McCallum, M. S. Keller, B. W. Herr, K. Börner, J. M. Spraggins, and N. Gehlenborg. Viv: Multiscale visualization of high-resolution multiplexed bioimaging data on the web. *Nat. Methods*, 19(5):515–516, 2022. doi: 10.1038/s41592-022-01482-7 7

[45] L. Marconato, G. Palla, K. A. Yamauchi, I. Virshup, E. Heidari, T. Treis, W.-M. Vierdag, M. Toth, S. Stockhaus, R. B. Shrestha, B. Rombaut, L. Pollaris, L. Lehner, H. Vöhringer, I. Kats, Y. Saeys, S. K. Saka, W. Huber, M. Gerstung, J. Moore, F. J. Theis, and O. Stegle. SpatialData: An open and universal data framework for spatial omics. *Nat. Methods*, pp. 1–5, 2024. doi: 10.1038/s41592-024-02212-x 2

[46] A. S. McNeal, R. L. Belote, H. Zeng, M. Urquijo, K. Barker, R. Torres, M. Curtin, A. H. Shain, R. H. Andtbacka, S. Holmen, D. H. Lum, T. H. McCalmont, M. W. VanBrocklin, D. Grossman, M. L. Wei, U. E. Lang, and R. L. Judson-Torres. BRAFV600E induces reversible mitotic arrest in human melanocytes via microRNA-mediated suppression of AURKB. *eLife*, 10:e70385, 2021. doi: 10.7554/eLife.70385 8

[47] A. Merouane, N. Rey-Villamizar, Y. Lu, I. Liadi, G. Romain, J. Lu, H. Singh, L. J. Cooper, N. Varadarajan, and B. Roysam. Automated profiling of individual cell–cell interactions from high-throughput time-lapse imaging microscopy in nanowell grids (TIMING). *Bioinformatics*, 31(19):3189–3197, 2015. doi: 10.1093/bioinformatics/btv355 2

[48] M. Meyer and J. Dykes. Criteria for Rigor in Visualization Design Study. *IEEE Trans. Visual Comput. Graphics*, 26(1):87–97, Jan. 2020. doi: 10.1109/TVCG.2019.2934539 9

[49] M. Meyer, T. Munzner, A. DePace, and H. Pfister. MulteeSum: A Tool for Comparative Spatial and Temporal Gene Expression Data. *IEEE Trans. Visual Comput. Graphics*, 16(6):908–917, 2010. doi: 10.1109/TVCG.2010.137 2

[50] T. Munzner. Rules of Thumb. In *Visualization Analysis and Design*. 2014. 5

[51] T. L. Nguyen, S. Pradeep, R. L. Judson-Torres, J. Reed, M. A. Teitell, and T. A. Zangle. Quantitative Phase Imaging: Recent Advances and Expanding Potential in Biomedicine. *ACS nano*, 16(8):11516–11544, Aug. 2022. doi: 10.1021/acsnano.1c11507 2

[52] C. Nobre, N. Gehlenborg, H. Coon, and A. Lex. Lineage: Visualizing Multivariate Clinical Data in Genealogy Graphs. *Transaction on Visualization and Computer Graphics*, 25(3):1543–1558, 2019. doi: 10.1109/TVCG.2018.2811488 2

[53] C. Nobre and A. Lex. OceanPaths: Visualizing Multivariate Oceanography Data. *Comput. Graphics Forum*, 2015. doi: 10.2312/eurovisshort.20151124 2

[54] C. Nobre, M. Meyer, M. Streit, and A. Lex. The State of the Art in Visualizing Multivariate Networks. *Comput. Graphics Forum (EuroVis)*, 38(3):807–832, 2019. doi: 10.1111/cgf.13728 2

[55] C. Nobre, M. Streit, and A. Lex. Juniper: A Tree+Table Approach to Multivariate Graph Visualization. *IEEE Trans. Visual Comput. Graphics (InfoVis)*, 25(1):544–554, 2019. doi: 10.1109/TVCG.2018.2865149 2

[56] J. S. Packer, Q. Zhu, C. Huynh, P. Sivaramakrishnan, E. Preston, H. Dueck, D. Stefanik, K. Tan, C. Trapnell, J. Kim, R. H. Waterston, and J. I. Murray. A lineage-resolved molecular atlas of C. elegans embryogenesis at single-cell resolution. *Science*, 365(6459):eaax1971, 2019. doi: 10.1126/science.aax1971 2

[57] A. Pandey, S. L'Yi, Q. Wang, M. A. Borkin, and N. Gehlenborg. GenoREC: A Recommendation System for Interactive Genomics Data Visualization. *IEEE Trans. Visual Comput. Graphics*, 29(1):570–580, 2023. doi: 10.1109/TVCG.2022.3209407 2

[58] A. J. Pretorius, I. A. Khan, and R. J. Errington. Cell lineage visualisation. *Comput. Graphics Forum*, 34(3):21–30, 2015. doi: 10.1111/cgf.12614 2

[59] A. J. Pretorius, I. A. Khan, and R. J. Errington. A Survey of Visualization for Live Cell Imaging. *Comput. Graphics Forum*, 36(1):46–63, 2017. doi: 10.1111/cgf.12784 2

[60] T. Saito, H. Miyamura, M. Yamamoto, H. Saito, Y. Hoshiya, and T. Kaseda. Two-tone pseudo coloring: Compact visualization for one-dimensional data. In *IEEE Symposium on Information Visualization (InfoVis)*, pp. 173–180. IEEE, Minneapolis, 2005. doi: 10.1109/INFVIS.2005.1532144 4

[61] I. Salvador-Martínez, M. Grillo, M. Averof, and M. J. Telford. CeLaVi: An interactive cell lineage visualization tool. *Nucleic Acids Research*, 49(W1):W80–W85, 2021. doi: 10.1093/nar/gkab325 2

[62] T. Schreck, D. Keim, and F. Mansmann. Regular TreeMap layouts for visual analysis of hierarchical data. In *Proc. 22nd Spring Conference on Computer Graphics*, SCCG '06, pp. 183–190. ACM, New York, 2006. doi: 10.1145/2602161.2602183 2

[63] H.-J. Schulz. Treevis.net: A Tree Visualization Reference. *IEEE Comput. Graphics Appl.*, 31(6):11–15, 2011. 2

[64] H.-J. Schulz, S. Hadlak, and H. Schumann. The Design Space of Implicit Hierarchy Visualization: A Survey. *IEEE Trans. Visual Comput. Graphics*, 17(4):393–411, 2011. doi: 10.1109/TVCG.2010.79 2, 6

[65] M. Sebesta, P. J. Egelberg, A. Langberg, J.-H. Lindskov, K. Alm, and B. Janicke. HoloMonitor M4: Holographic imaging cytometer for real-time kinetic label-free live-cell analysis of adherent cells. In *Quantitative Phase Imaging II*, vol. 9718, p. 971813. International Society for Optics and Photonics, 2016. doi: 10.1117/12.2216731 2

[66] M. Sedlmair, M. Meyer, and T. Munzner. Design Study Methodology: Reflections from the Trenches and the Stacks. *IEEE Trans. Visual Comput. Graphics*, 18(12):2431–2440, 2012. doi: 10.1109/TVCG.2012.213 7

[67] J. Seo and B. Shneiderman. Interactively exploring hierarchical clustering results [gene identification]. *Computer*, 35(7):80–86, 2002. doi: 10.1109/MC.2002.1016905 2

[68] A. Somarakis, M. E. Ijsselsteijn, S. J. Luk, B. Kenkhuis, N. F. de Miranda, B. P. Lelieveldt, and T. Höllt. Visual cohort comparison for spatial single-cell omics-data. *IEEE Trans. Visual Comput. Graphics*, 27(2):733–743, 2021. doi: 10.1109/TVCG.2020.3030336 2

[69] T. Tekusova and T. Schreck. Visualizing Time-Dependent Data in Multivariate Hierarchic Plots - Design and Evaluation of an Economic Application. In *12th International Conference Information Visualisation*, pp. 143–150, 2008. doi: 10.1109/IV.2008.51 2

[70] J. Troidl, C. Cali, E. Gröller, H. Pfister, M. Hadwiger, and J. Beyer. Barrio: Customizable Spatial Neighborhood Analysis and Comparison for Nanoscale Brain Structures. *Comput. Graphics Forum*, 41(3):183–194, 2022. doi: 10.1111/cgf.14532 2

[71] J. Troidl, S. Warchol, J. Choi, J. Matelsky, N. Dhanyasi, X. Wang, B. Wester, D. Wei, J. W. Lichtman, H. Pfister, and J. Beyer. ViMO - Visual Analysis of Neuronal Connectivity Motifs. *IEEE Trans. Visual Comput. Graphics*, 30(1):748–758, 2024. doi: 10.1109/TVCG.2023.3327388 2

[72] A. Van Der Ploeg. Drawing non-layered tidy trees in linear time: DRAWING NON-LAYERED TIDY TREES IN LINEAR TIME. *Softw.: Pract. Exper.*, 44(12):1467–1484, 2014. doi: 10.1002/spe.2213 7

[73] T. Walter, D. W. Shattuck, R. Baldock, M. E. Bastin, A. E. Carpenter, S. Duce, J. Ellenberg, A. Fraser, N. Hamilton, S. Pieper, M. A. Ragan, J. E. Schneider, P. Tomancak, and J.-K. Hériché. Visualization of image data from cells to organisms. *Nat. Methods*, 7(3):S26–S41, 2010. doi: 10.1038/nmeth.1431 2

[74] S. Warchol, R. Krueger, A. J. Nirmal, G. Gaglia, J. Jessup, C. C. Ritch, J. Hoffer, J. Muhlich, M. L. Burger, T. Jacks, S. Santagata, P. K. Sorger, and H. Pfister. Visinity: Visual Spatial Neighborhood Analysis for Multiplexed Tissue Imaging Data. *IEEE Trans. Visual Comput. Graphics*, pp. 1–11, 2022. doi: 10.1109/TVCG.2022.3209378 2

[75] Y. Yang, F. Dadgostar, C. Sanderson, and B. C. Lovell. Summarisation of surveillance videos by key-frame selection. In *2011 Fifth ACM/IEEE International Conference on Distributed Smart Cameras*, pp. 1–6. IEEE, Ghent, Belgium, 2011. doi: 10.1109/ICDSC.2011.6042925 3

[76] J. Zhang, J. Griffin, K. Roy, A. Hoffmann, and T. A. Zangle. Tracking of Lineage Mass via Quantitative Phase Imaging and Confinement in Low Refractive Index Microwells, 2024. doi: 10.1101/2024.03.27.587085 8

[77] Y. Zhang, R. L. Belote, M. A. Urquijo, M. M. K. Hansen, M. Hejna, T. E. Moustafa, T. Liu, D. Lange, F. Vand-Rajabpour, M. Chang, B. K. Lohman, C. Stubben, X. Zhang, L. S. Weinberger, M. W. VanBrocklin, D. Grossman, A. Lex, R. Kulkarni, T. Zangle, and R. L. Judson-Torres. Bidirectional interconversion between mutually exclusive tumorigenic and drug-tolerant melanoma cell phenotypes, 2023. doi: 10.1101/2020.08.26.269126 3, 8, 9