




NORBERT LINDOW¹, FLORIAN N. BRÜNIG², VINCENT J.
DERCKSEN³, GUNAR FABIG⁴, ROBERT KIEWISZ⁵,
STEFANIE REDEMANN⁶, THOMAS MÜLLER-REICHERT⁷,
STEFFEN PROHASKA⁸


Semi-automatic Stitching of Serial Section Image Stacks with Filamentous Structures


¹  0000-0001-5143-2573

²  0000-0001-8583-6488

³ Zuse Institute Berlin, Germany

⁴  0000-0003-3017-0978

⁵  0000-0003-2733-4978

⁶  0000-0003-2334-7309

⁷ Technische Universität Dresden, Germany

⁸ Zuse Institute Berlin, Germany

Zuse Institute Berlin
Takustr. 7
14195 Berlin
Germany

Telephone: +49 30-84185-0
Telefax: +49 30-84185-125

E-mail: bibliothek@zib.de
URL: <http://www.zib.de>

ZIB-Report (Print) ISSN 1438-0064
ZIB-Report (Internet) ISSN 2192-7782

Semi-automatic Stitching of Serial Section Image Stacks with Filamentous Structures

Norbert Lindow¹, Florian N. Brünig², Vincent J. Dercksen³,
Gunar Fabig⁴, Robert Kiewisz⁵, Stefanie Redemann⁶,
Thomas Müller-Reichert⁷, Steffen Prohaska⁸


December 10, 2019


Abstract

In this paper, we present a software-assisted workflow for the alignment and matching of filamentous structures across a stack of 3D serial image sections. This is achieved by a combination of automatic methods, visual validation, and interactive correction. After an initial alignment, the user can continuously improve the result by interactively correcting landmarks or matches of filaments. This is supported by a visual quality assessment of regions that have been already inspected and, thus, allows a trade-off between quality and manual labor.


The software tool was developed in collaboration with biologists who investigate microtubule-based spindles during cell division. To quantitatively understand the structural organization of such spindles, a 3D reconstruction of the numerous microtubules is essential. Each spindle is cut into a series of semi-thick physical sections, of which electron tomograms are acquired. The sections then need to be stitched, that is sections need to be non-rigidly aligned, and microtubules need to be traced in each section and connected across section boundaries. Experiments led to the conclusion that automatic methods for stitching alone provide only an incomplete solution to practical analysis needs. Automatic methods may fail due to large physical distortions, a low signal-to-noise ratio of the images, or other unexpected experimental difficulties. In such situations, semi-automatic validation and correction is required to rescue as much information as possible in order to derive biologically meaningful results despite some errors related to data collection.


These are difficult tasks, because the correct stitching is visually not obvious due to the number of microtubules (up to 30k) and their dense spatial arrangement. Furthermore, a naive inspection of each microtubule is too time consuming. In addition, interactive visualization is hampered by the size of the image data (up to 100 GB). Based on the requirements of


¹  0000-0001-5143-2573

²  0000-0001-8583-6488

³ Zuse Institute Berlin, Germany

⁴  0000-0003-3017-0978

⁵  0000-0003-2733-4978

⁶  0000-0003-2334-7309

⁷ Technische Universität Dresden, Germany

⁸ Zuse Institute Berlin, Germany

our collaborators, we present a practical solution for the semi-automatic stitching of serial section image stacks with filamentous structures.

Introduction

Serial section imaging can be used to acquire images of 3D structures that are too thick to image them directly. A specific application is serial section electron tomography, where each section is a 3D tomogram [1]. During serial sectioning, the specimen is physically cut into a series of blocks, where each block is imaged separately. Afterwards, the 3D tomograms need to be digitally stitched together. Depending on possible deformations during the imaging process, this often requires a non-linear alignment. Corresponding structures in neighboring sections can be identified either based purely on image data or on segmented features. In this paper, we present a software-assisted workflow to semi-automatically stitch serial section tomography data using segmented filament structures. Two connected problems need to be solved: (1) the alignment problem, where the serial sections are transformed such that they fit together; (2) the matching problem, where the filament structures are combinatorially connected between neighboring sections.

A fully automatic approach to stitch such tomograms is often problematic for two main reasons. First, the data quality may be poor due to the size of the tomograms and inaccuracies during specimen preparation or microscopy image acquisition. The sections may suffer from physical damage: they may have been deformed or ruptured, causing the alignment to fail. Second, visual validation and correction of the stitching result is difficult. Interactively inspecting each filament can require a prohibitive amount of time. With limited time, it is not obvious which subset to choose for inspection. There is also no mechanism to interactively correct the alignment or filament matching if necessary, or to integrate user-provided information into the automatic matching process to improve it. If for some reason the automatic stitching failed between two sections, often the full stack cannot be used anymore. Thus, in absence of options to manually improve aligned data sets of insufficient quality, the biological value of these data sets is limited.

In this paper, we present a software-assisted workflow to address these issues, with the following key contributions:

- A user-controlled workflow to solve the stitching problem that integrates automatic alignment and matching. The automatic algorithms take user input into account to quickly generate and continuously improve the result.
- The *Serial Section Aligner*, a new software tool with multiple-view visualizations and interactions designed to support this workflow. A user can inspect both the images and the segmented filaments in order to validate and correct the alignment and matching.
- A new visualization method to monitor which section regions have been visually inspected by the user.

The workflow is demonstrated on a specific biological application: serial section electron tomography of both mitotic and meiotic microtubule-based spindles of high-pressure frozen and plastic-embedded material [2].

Serial Section Electron Tomography of Spindles

The segregation of chromosomes in either mitosis or meiosis is an essential mechanism in cell division. The correct segregation of chromosomes is indispensable for a cell to survive. Failures can lead to defects for the emerging daughter cells as well as the entire organism. In general, missegregation is a major cause of aneuploidy and cancer. Therefore, a detailed understanding of the mechanism of chromosome segregation is of significant biological importance.

The microtubule bipolar spindle is a key structure in both mitotic and meiotic cell division. The different stages of mitotic cell division are illustrated in Fig 1A. During prophase (2) the centrosomes (green) start to move towards opposite sites of the cell, and a bipolar spindle mainly consisting of microtubules (MTs) begins to form. At this point, the DNA is visible as chromosomes. The chromosomes then move towards the center of the cell at prometaphase (3). The MTs of the spindle align the condensed chromosomes on the metaphase plate (4) and then segregate the chromatids to the forming daughter cells in anaphase (5). Finally, the mitotic spindle is disassembled, and the nuclear membranes around the daughter chromosomes are reformed in telophase (6). Quantitative analysis of MT dynamics is crucial for understanding spindle assembly and organization. Our collaborators aim to investigate this process using 3D reconstructions of spindle specimens at different phases of cell division and for a comparison of wild-type and mutant samples [3, 4].

In order to create a 3D reconstruction of the spindle at sufficiently high resolution to be able to identify the MTs, cells are prepared for plastic embedding and then physically sectioned for imaging by electron tomography at 5–8 nm resolution [5], resulting in up to 30 sections with up to 5 GB per section, Fig 1B-2. The spindle geometry is then reconstructed in two steps. First, the MTs are segmented in each section separately using template matching. This is followed by an automatic tracing algorithm with a user verification [6], Fig 1B-3. In the second step, the segmented MTs are used to automatically stitch the serial sections to a single stack [7]. Due to the local deformations during electron tomography, a non-linear alignment is required that often cannot be immediately determined due to the lack of obvious image features. Hence, the sections are aligned by applying a variant of the coherent point drift algorithm (CPD) to the segmented MTs. For each section interface, the algorithm uses the MT ends of the two neighboring sections with the 3D directions at the MT ends and returns 2D landmarks as pairs of corresponding points. The MTs and tomograms are then warped using moving least squares based on the landmarks. Finally, a matching algorithm is applied on the transformed MTs to connect them over section boundaries, Fig 1B-4.

The described stitching was designed and implemented as a fully automatic pipeline [7], which allowed only a final verification when the complete stack was built. Furthermore, a visual verification of the complete stack is difficult. Only small manual corrections seem realistic for the full stack. If the alignment completely failed for one section interface, for example, it was impossible to create a meaningful stack.

Although the following semi-automatic solution has been developed in the context of MT reconstruction from stacks of electron tomograms, it is in principle suitable for other problems where stacks of 3D volumes containing filamentous structures need to be aligned. The filaments are explicitly represented

and used throughout the entire process to achieve high reconstruction quality with consistent filament geometry accross sections, which seem unrealistic with generic image-based registration approaches.

Requirements

The goal is to reconstruct filamentous structures in a serial section image stack. The following requirements were identified for the stitching process in cooperation with our collaborators.

The overall pipeline should be semi-automatic. The user should be able to influence all steps of the stitching process. In addition, the pipeline should be flexible such that the user can change the alignment or matching at any time. In the worst case, the user should be able to perform the whole stitching manually. In the best case, the user should be able to apply the stitching completely automatically. For cases of medium quality, minor manual interactions should support the automatic algorithms to rapidly increase the quality of the results. The whole pipeline should be supported by visualizations to verify the current state and offer options to interactively correct the state. Finally, the application should be able to interactively handle all relevant data sizes, which was up to 30 tomograms with 5 GB each in our specific case.

Based on these requirements, we decided that the alignment and matching would be first solved locally for each section interface, that is the cut between two neighboring sections. Finally, the full stack would be constructed from the local solutions. The requirements for stitching a single section interface can be summarized as follows:

- The two sections of the interface should be interactively visualized with the current local alignment and matching such that it is quickly possible to detect errors by inspection.
- It should be possible to set or change the alignment with manual landmarks through simple interactions; the user should be able to completely discard the automatic alignment.
- The automatic alignment should use the manual landmarks for a better initialization.
- The user should be able to change the current matching by adding or removing matches manually. Manual matches should influence the re-computation of the automatic matching. Furthermore, the user should be able to confirm automatic matches or unmatched MT ends, which will from then on remain in the confirmed state.
- In contrast to the fully automatic pipeline, only the matched endpoints should impact the automatic alignment, instead of all filament ends, in order to allow controlling consistent microtubule geometry accross sections.
- The application should give immediate visual feedback about the incremental matching progress.

1 Related Work

Transmission electron microscopy/tomography is an important technique for imaging biological structures at molecular resolution [1]. To acquire images that cover a large sub-volume of a cell, one approach is serial sectioning combined with tomography of semi-thick sections. Each section can furthermore be virtually subdivided into tiles that are acquired separately to increase the field of view.

Tools and methods are required to reconstruct the complete 3D image from the tiles and sections. Typically, this is done in reverse order. First, the tiles are aligned to reconstruct a single section, called mosaicking. Second, the serial sections are combined to a complete stack. Several automatic methods were previously proposed for both tasks. Especially serial section electron microscopy for neuronal data has been investigated [8, 9]. 3D electron tomography typically has a lower signal-to-noise ratio with less contrast than 2D electron microscopy and requires an additional flattening phase for each section. Weber et al. [7] proposed an automatic workflow to align serial section tomograms that contain MTs.

Automatic methods accelerate the reconstruction of large volumes and usually operate successfully for images of expected quality. Automatic methods, however, can fail for specific cases due to unexpected data quality or underrepresented structures. In such situations, interactive tools are helpful to solve these specific cases and to allow the user to control the quality of the result. One of the most widely used semi-automatic software packages for electron tomography reconstruction is IMOD [10]. It is used in many laboratories for semi-automatic mosaicking and flattening of tomograms.

A tool that focuses on the alignment of serial sections of electron microscopy images was proposed by Anderson et al. [11]. The tool, named *ir-tweak*, provides an easy way to place landmarks with a direct real-time preview of the resulting non-linear warping using thin plate splines. We transferred this design to the alignment of serial section tomograms containing filamentous structures. In contrast to thin plate splines, however, we use moving least squares for the warping.

Berlanga et al. [12] proposed a workflow for reconstructing 3D images from serial section confocal microscopy for the mouse brain. Their workflow includes the flattening of the sections using manual landmark generation with IMOD as well as semi-automatic alignment of the section with *ir-tweak*.

In 2012, Cardona et al. [13] proposed a tool for investigating neural circuits from serial section microscopy images. The alignment is done completely automatically. The focus of their work is the semi-automatic interaction with the image data.

To explore large connectome data sets, Beyer et al. [14] proposed a framework that enables petascale volume visualization. The complete mosaicking and sectioning is done in real time based on an automatically pre-computed transformation. As for Cardona et al., the tool is not designed to manually edit the transformation but to segment and explore neurites and synapses in the data. More recently, the same authors proposed an advanced tool to organize and inspect the whole segmentation process [15].

A tool for a related application area, called *Filament Editor*, was proposed by Dercksen et al. [16]. It allows reconstructing 3D neuronal networks based

on serial optical microscopy sections. The tool provides 3D visualization of the networks and image volumes as well as 2D slices and allows users to manually edit the networks. The serial section alignment is computed automatically with a linear transformation. The networks that can be handled are sparse and small compared to the microtubules data sets that are the focus here.

2 Application Software

We implemented the tool as two modules in the visualization software Amira [17]. The first module is a data class that manages the input data and the stitching state. The second module implements the visualizations and editing operations of the stitching workflow.

2.1 Serial Section Stack

The data class is called *serial section stack*. Let n be the number of serial section tomograms, which are denoted by T_1, \dots, T_n . We assume that each tomogram has been pre-processed, that is distortions along the thin section side, which is usually the z -direction, have been removed in a flattening pre-processing step, followed by a cropping step. Furthermore, the filaments have been segmented and verified for each tomogram. The set of filaments F_i of tomogram T_i is a set of piecewise linear curves $f \in F_i$ with $f = \{p_1, \dots, p_k\}$ and $p_j \in \mathbb{R}^3$. We assume that the stack is always built in z -direction. We call the x - y -plane that displays the tomogram data for a certain z -value a z -slice.

A new stack can be created by adding tomograms and corresponding filament data sets with a file dialog. For each of the $n - 1$ interfaces between neighboring sections, the stitching state is stored as a set of landmarks on the section boundaries and a matching that represents corresponding filament ends. A button allows the user to reverse the section order.

2.2 Serial Section Alignment

The stitching process is implemented as a second module, called *serial section aligner*. The module provides two modes. The first mode, called *alignment mode*, allows the user to manually add, modify, and remove landmarks. The second mode, called *matching mode*, enables the automatic computation of a matching in combination with manual editing and verification. In addition to the mode-dependent interactive visualization, which consists of four sub-views, the module has several GUI parameters (Fig 2A). The first part of the GUI is independent of the mode. It allows selecting the currently investigated section interface, adjusting visualization parameters for the interface, changing the position of the two z -slices for the neighboring tomograms, and adjusting the scale of the filament radii. The user can also choose to control quality parameters, which have a major effect on the performance. The second part of the GUI depends on the selected mode, as described below. The third part allows the user to adjust and create the full stack as the final step of the stitching process.

Alignment Mode

This mode is for adjusting the alignment of a pair of tomograms for a selected interface. The four views in the visualization window are used in the following way (Fig 3). The top left view displays a z -slice of the upper tomogram of the selected interface. Accordingly, the bottom left view displays a z -slice of the lower tomogram. The user can move and zoom simultaneously in both slices to navigate. Additionally, the user can change the positions of both z -slices with two sliders (Fig 2A). The two sliders are synchronized per default; when the top z -slice is decreased by the user, the bottom one is automatically increased and the other way around. The views allow the user to identify corresponding features in the tomograms, such as vesicles, chromosomes, or membranes. For these features, the user can add, move, or delete manual landmarks with mouse interactions in the two left views. A landmark consists of a pair of 2D points, one point for each tomogram. The points are placed such that they mark corresponding biological structures. They are visualized as red circles on the z -slices. The radii of the circles can be scaled by a slider. During this placement, a rigid transformation is computed based on the current landmarks. The transformation globally rotates and translates the top slice, ignoring distortions, such that corresponding regions are displayed for the top and bottom tomogram during navigation (panning and zooming).

In the top right view, an overlay of the selected z -slices is visualized by using transparency in combination with two different colors: blue for the top slice and yellow for the bottom slice. Based on the interactions in the left views, the overlay view is updated in real time to preview the warping based on the current landmarks. To do so, the upper tomogram is warped in real time such that for each landmark, the corresponding locations are mapped onto each other. This allows the user to interactively evaluate the current alignment and to iteratively improve it by adding more landmarks or by modifying existing landmarks.

In addition to the tomogram data, the filaments within the slices are visualized in all three views. Filaments of the bottom tomogram are displayed in yellow and the filaments of the top tomogram in blue.

The forth view, in the bottom right, shows an abstract illustration of the whole stack and the selected interface with the pair of tomograms. The user can change the interface with a mouse click onto the illustration. Again, the same colors are used, yellow for the bottom and blue for the top tomogram. Next to the stack, a vertical histogram is shown. Each bin of the histogram shows the density of the unmatched filament ends for 25% of a tomogram in z -direction. At the beginning, the histogram usually shows higher densities at the interface boundaries, because the filaments are not yet matched. During the stitching process these differences decrease, as more and more filament ends are matched. The plot helps the user to evaluate the overall quality, but also to identify the most critical interfaces that need further investigation. A high density of unmatched ends at a section boundary compared to the ends within the section indicates an incomplete matching.

Apart from manual landmarks, automatic landmarks (yellow) can be generated from matched filaments, which is described in the following section. The user can also remove and manipulate these landmarks. If the automatic landmark placement is activated, these landmarks will, however, be updated by

changes in the matching.

Matching Mode

In this mode, the user defines the matching between the filaments of the two tomograms. This is supported by four views and the possibility to automatically compute the matching in combination with manual editing (Fig 4).

In the top right, again an overlay of the z -slices is visualized with the filaments. The filaments are colored according to the matches, where automatic matches are depicted in different shades of red and manual ones in shades of green. The colors are taken from ColorBrewer.org [18]. Unmatched filaments are visualized in black. Because a filament can be matched at both ends, the color fades out from one end to the middle of the filament to avoid ambiguous coloring. In addition to the coloring, a match between two filaments is visualized by a line, which also fades out accordingly. To keep track of which filament corresponds to which tomogram, a colored border around each filament is drawn. Again, a blue border is used for the filaments of the top tomogram and a yellow for the ones of the bottom tomogram. The user can select two filament ends of the two sections with the mouse to either add a manual matching or to remove it.

To facilitate the matching decisions, the overlay view is supported by a side view on the bottom left and a 3D view on the top left. During navigation, the three views are simultaneously updated. To keep track of the side view, two lines are displayed in the overlay view that illustrate the position and orientation of the side view. The user can select an arbitrary filament of the bottom tomogram in the overlay view, and the side view is immediately aligned such that this filament is centered and contains the tangent of the filament. The side view enables the user to evaluate the slope of matched or unmatched filaments. It also helps detecting distortions that prohibit an automatic matching and help defining manual matches for such cases. Similarly to the lines in the overlay view that depict the positions of the side view, two lines in blue and yellow depict the position of the z -slices. The 3D view shows all MTs of a local region around the center of the overlay view. In addition to zooming and panning, the user can rotate the scene in this view. To avoid visual clutter, the region is cut by a sphere with a user selected radius. Similarly to the side view, the 3D view shows a part of the tomogram within the cutting sphere as a transparent slice. The 3D view helps to investigate the shape of multiple filaments at once.

The user can automatically compute or recompute the matching at any time based on the current alignment. Currently, the matching is computed using the algorithm described by Weber et al. [7]. However, this part works as a black box for any suitable filament matching algorithm. The input of the algorithm are the 3D positions of the filament ends with approximated 3D directions of the two neighboring sections. Note that the directions and positions are transformed based on the current alignment. The region for the selected ends is defined by m percent of the top part of the bottom section and m percent of the bottom part of the top section, where m is defined by the user. Furthermore, the algorithm receives information about all matched filament ends that should not be touched anymore. If the user checks ‘automatic landmarks’ in the GUI (Fig 2A), the ends of matched filaments will be used to generate automatic landmarks for the alignment of the sections. A validation stepper is embedded in the GUI and is

accessible via keyboard shortcuts. It allows the user to step through all filament ends in the matching region of the bottom section. The stepper automatically centers the views to the currently selected filament end. The user can confirm the current state of a filament end, whether it is matched or not. The stepper will not display confirmed ends again. All filament ends with manual matches are confirmed by default. When all filament ends of the bottom tomogram have been confirmed, the states of the ends of the top tomogram are implicitly also confirmed. For this reason, it is sufficient to observe the filament ends on only one side of the interface. Due to time restrictions, users are often not willing to confirm all filament ends, but only a fixed number such as 100–200. To avoid that the stepper displays only filament ends in one region, the order of the ends is shuffled randomly. In addition to the stepping, the user can directly select a filament end of the bottom tomogram in the overlay view and confirm its state. The GUI always displays the number of confirmed and open ends.

The fourth view is used to evaluate the current state of the matching process. It shows the z -slice of the bottom tomogram, subdivided into Voronoi regions of the filament ends in the matching region. Regions that correspond to filament ends for which the user confirmed the state are displayed bright. Automatically matched ends that were not explicitly confirmed by the user are displayed less bright. The remaining regions, containing all unmatched unconfirmed filament ends, are displayed in dark. The map always displays the same view for the bottom section as in the overlay view. By selecting a region in this map, the stepper jumps automatically to the corresponding microtubule end. This map helps to evaluate the automatic matching and to steer the manual investigations. Especially for large data, it helps to identify regions in the tomogram that need further attention.

Within the fourth view, again the end point density for the current interface is shown. The histogram consists only of four bins, two for the upper 50% of the bottom tomogram and two for the lower 50% of the top tomogram. The leftmost histogram shows the current state, while the others show the temporal evolution. With this, the user can quickly evaluate the impact of the manual editing.

Stack Construction

The last area of the GUI is dedicated to the creation of the full stack at the end of the stitching process. One section is selected as the base section, which is not warped. The sections above and below the base section are warped based on the landmarks of the corresponding interfaces. The warped tomograms are resampled into a single tomogram volume. The warping is done similarly for the filaments. Matched filaments are joined over the section interfaces. Each point of a filament receives an integer attribute that stores the original section ID of the point. If the quality of the tomograms and the distribution of the filaments is more or less uniform over the full stack, the middle section should be ideal as a base section to avoid accumulating the warping error over more interfaces than necessary. Otherwise, one should select a section close to the middle of the stack with the best quality and filament distribution.

A checkbox allows the user to create only the stack of the filaments without the aligned tomograms, which is much faster. Choosing a lower resolution for the x - and y -direction of the tomograms is another way to increase speed and

reduce memory requirements. Both options are helpful to quickly generate a preview of the stack or create results on systems that have too little RAM to merge the tomograms at full resolution.

2.3 Recommended Workflow

The recommended overall workflow is as follows (Fig 2B). After creating the *serial section stack* object and the detection of the correct stack order, the section interfaces will be processed one after another. For each interface, the sections should be sufficiently aligned and the filaments matched such that it should not be necessary to revisit the interface. This is recommended, because frequent switches between section interfaces can be time consuming due to repeated tomogram loading. If the quality of the tomograms is good and the filaments are distributed well over the whole tomograms, the user should directly switch to matching mode and check the automatic computation of the matching. If the matching and alignment looks good, the user should step over a fixed number of filament ends and confirm or correct them. Finally, the user can add a few manual landmarks in regions without filaments to improve the alignment if necessary.

If the automatic matching failed completely, the user should clear the matching. Now, there are two possibilities to continue, which can also be combined. First, the user can set a few manual landmarks in alignment mode and retry the automatic matching. Second, the user can manually match a few filaments, which also results in new landmarks, and retry the automatic matching. Usually, the second option is only reasonable if the sections are almost aligned. As soon as the automatic matching seems to produce reasonable results, an iterative process starts with correcting or adding manual matches in combination with an automatic re-computation. Usually, small manual corrections help to improve the whole region in the neighborhood with the automatic matching. It is recommended to look for the same unique patterns of filament ends in both sections by moving the z -slices. An example is shown in Fig 7A, where the automatic matching failed locally, and the placement of two manual matches helped to improve the automatic matching for the complete region. Finally, the same procedure is followed as described above for a direct successful automatic matching.

In some situations, especially when the direction of the filaments is highly distorted, it can be useful to either completely deactivate the automatic landmark placement by the checkbox in the GUI or to forbid the landmark computation for a certain filament match. The latter can be done with a keyboard shortcut for the selected matched filament end. The already placed automatic landmarks can be removed in alignment mode. In case the automatic landmark placing is disabled, mainly modifications of manual landmarks can improve the automatic matching.

3 Visualization Methods

This section provides technical details about the interactive visualization implementation.

3.1 Stack Data Structure

As described in Sec 2.1, the serial section data is handled in a stack data structure. Let n be the number of sections. Then the stack stores n tomograms T_i with the n corresponding filament data sets F_i . The data sets are stored in the file system. The stack data structure loads tomograms and filaments on demand and hands them over to the aligner module, which is then responsible for the cleanup. During alignment and matching, only 2 tomograms are loaded into RAM at a time.

For each of the $n - 1$ section interfaces between consecutive tomograms, a set of landmarks and a matching is stored. As briefly described above, each landmark consists of a pair of 2D points in x - y -space, one point for each of the neighboring sections. The points mark corresponding locations on the two sides of an interface and define an x - y -transformation that warps one section into the space of the neighboring one such that the tomograms and filaments are aligned. Because the sections are thin with a flat boundary in z -direction after the pre-processing step, warping is only defined as an x - y -transformation, ignoring z . In addition to the pair of points, each landmark has a type attribute that stores the source of the landmark. The source can be *manual* or *automatic*. As described earlier, these types will be colored in red and light yellow, respectively.

A matching is stored as a set of quadruples. A match between F_i and F_{i+1} is stored as $(a, 1, b, 0)$, where the end of filament $a \in F_i$ is matched with the start of filament $b \in F_{i+1}$. Similar to the landmarks, each match has a type attribute that is either *manual* or *automatic*.

Filament end states are stored for the bottom section of each interface to maintain the matching state. A filament end state can be either *confirmed* or *undefined*.

3.2 Slice Rendering

The rendering of slices within the tomograms is essential for the visual assessment of the filament alignment and matching. Our tool allows interactively visualizing top- and side-view slices for original and for warped tomograms.

First, we consider the case without warping. The cost for rendering a slice that is computed in tomogram space, that is the slice is sampled at tomogram resolution, scales with the tomogram size. It can become too expensive to achieve interactive frame rates. The screen, however, has a limited number of pixels. In most cases, either only a part of the slice is displayed, or it is so far away from the camera that the size of a voxel is much smaller than a pixel. Hence, small details are not visible, and it would be inefficient to compute a slice at full resolution in tomogram space. Instead, we compute slices in view space. In the first step, we compute for each pixel of the view the 3D position within the slice. This is done by computing the view ray through the pixel and detecting the intersection of the ray with the slice plane. Additionally, it is checked if the intersection point is located outside the bounding box of the tomogram. In the next step, we evaluate for each point inside the tomogram the intensity value using trilinear interpolation. Both steps are performed on the CPU using OpenMP, because tomograms may not fit into GPU RAM. Finally, a screen space filling quad is rendered that is textured with the computed intensity values. With this approach, the number of memory accesses into the tomogram is

limited by the number of pixels of the screen. Although we observed that rendering is interactive even for large tomograms and many cache misses, the user can switch between different slice qualities. For high quality, the full resolution is considered, for medium, half the view resolution, and for low, only one fourth of the view resolution. This can further increase the interactivity, especially on 4K screens.

Consider now the warped case for the overlay views of the tomograms. Again the 3D position of the slice for each pixel is computed in the space of the bottom tomogram, but without the check whether a point is located inside or outside the tomogram. To get the correct intensity values of the top tomogram in the space of the bottom tomogram, the positions need to be transformed into the space of the top tomogram. This is done by the moving least squares approach by Schaefer et al. [19]. Depending on the number of landmarks, it can take too long to render slices interactively, especially for changing landmarks. However, typically the number of landmarks is much lower than the number of voxels within the slice. Furthermore, local warping results cannot be observed if the slice is far away from the user. For this reason, we create a 2D grid, again only for screen space. For each grid point, the corresponding 3D point on the slice is detected. Then, only the grid points are warped. For all points in a grid cell, bilinear interpolation of the warped cell is applied to compute the warped position (Fig 5). This accelerates the warped slice rendering such that it can be interactively updated during landmark or slice changes. Additionally, it creates a kind of level of detail: the correctness of the warping increases the closer the user zooms into the slice. Since the warping affects only the x - y -space of the tomogram, it is sufficient to use a 1D grid for side views.

In the original implementation of the moving least squares approach, all landmarks are used to transform a point. However, since the influence of a landmark decreases with increasing distance, it is sufficient to detect the closest k landmarks to transform a point. To do this in an efficient way, all landmarks are stored in a grid data structure. The restriction to the k nearest landmarks accelerates the rendering of the warped tomogram slices and filaments.

3.3 Landmark Rendering

Each landmark is visualized in the alignment mode as a circle on the z -slice. The circles have a constant size in screen space with a scaling parameter. This seems to be superior to object-space radii, because the visible circle size remains constant across zoom levels, which simplifies UI interaction. For each circle, a single vertex is generated that is then extended to a quad in the geometry shader on the GPU. The center of the quad is the center of the landmark, and the sides are twice as long as the radius of the circle. Then, in the fragment shader, the corresponding position on the quad is inserted into the implicit circle equation to check whether the fragment is inside or outside the circle. All fragments inside the circle are colored according to the landmark type. In addition, the transparency of the color is increased for fragments close to the center of the circle. This allows the user to see the image feature that is related to the landmark.

3.4 Filament Rendering

As mentioned in Sec 2.1, each filament is represented as a piecewise linear curve, given by a set of points p_1, \dots, p_n . We render a sphere at each position p_i and a cylinder between two position p_i and p_{i+1} . Note that both, the spheres and the cylinders, have the same radii. To display the filaments within the selected slices in the tomogram, only the intersection of the spheres and cylinders with the slice is visualized. To avoid an expensive triangulation of the spheres and cylinders with additional intersection computations, the rendering is based on the ray casting of quadrics proposed by Sigg et al. [20]. Each sphere is represented initially as a single 3D vertex. A geometry shader is used to create a quad from this vertex that encloses the sphere after projection. The ray casting is performed in the fragment shader. For the 2D views of the filaments, only the intersections of the cylinders with the slice are considered. Similarly, for each cylinder, one 3D vertex is created for the start point and one 3D vertex for the end point. Again in the geometry shader, a quad is created that encloses the cylinder after projection. Finally, the ray casting with the intersection test is performed in the fragment shader.

To quickly update the filament visualization of the warped tomogram in the overlay view, all spheres and cylinders that intersect the slice are detected in a first step. Then, only these spheres and cylinders are warped and visualized.

Both, filaments and landmarks, are rendered first in a frame buffer. During the rendering on the slice, a shader is used to apply anti-aliasing to the boundaries. To do so, for each fragment at the boundary, the tangent line is computed based on an analysis of the neighboring fragments in the four main directions. The line is used to compute a blending value for the fragment. Furthermore, for close-up views on the filaments or landmarks, the line is used to create a pixel wide anti-aliased black silhouette. This procedure dramatically increases the visual quality and helps to distinguish between slice data and filaments or landmarks.

3.5 Filament End State Map

In matching mode, a map depicts which filament ends are already confirmed, which ends are automatically matched but not confirmed, and which ends are unmatched and not confirmed (Fig 6). This map is generated and updated in the following way.

First, a discrete 2D scalar field is created that has the same boundary in the x - and y -directions as the bottom tomogram. Then for each sample position of the scalar field, the closest distance to all filament ends in the matching region of the bottom section is computed. The identifier of the closest filament end is stored at this position in the scalar field. Hence, the scalar field is a discrete representation of the Voronoi diagram of the filament ends in the matching region. Note that this map needs to be updated only when the user changes the matching region.

Then the z -slice of the bottom section is rendered as described above. However, during the detection of the intensity value in the tomogram for a pixel, additionally the closest filament end is detected by the pre-computed Voronoi diagram. Then the state of the end is used to modulate the brightness of the intensity value. The intensity values are darkened for all unconfirmed unmatched

ends and brightened for all confirmed ends. An illustration of the filament end state map is displayed in Fig 6.

4 Results

This section presents two types of results. First, we demonstrate that the proposed tool solves the matching and alignment of serial section stacks for data sets of different size, number, and distribution of microtubules. Second, we show that the performance of the tool is sufficient for the relevant use cases.

4.1 Matching and Alignment Examples

So far, at least 25 full spindle data sets were assembled by 5 different users (Table 1). The data sets were obtained from different specimens, spindle types, and stages of cell division, as well as from both wild-type and mutant samples. In Fig 8, four of these spindles are shown as reconstructed tomogram stacks and stitched MTs. They demonstrate the diversity of microtubule-based spindles.

In the following three paragraphs, we describe in some detail the most challenging data sets. For the first two, the users had to tackle experimental problems, such as damaged sections or poor image quality. The final example is the largest spindle that has been reconstructed so far. We used it as a benchmark for our system.

Mitotic Spindle Prometaphase

In this phase of cell division, centrosomes nucleate numerous microtubules, which enter through the fragmenting nuclear envelope in order to align the chromosomes at the center of the cell. The sample stack contains a region around the nucleus and consists of 7 sections with 300–500 MTs per section (Table 1 T0345-11). Due to the field of view with the nucleus in the center, most of the MTs are close to the image boundaries in x - y . The original, fully automatic workflow failed (Fig 7C). None of the section interfaces could be aligned. The reason was probably the uneven distribution of the microtubules. With a few manual landmarks, often only 3–5, the automatic matching worked quite well. If the global transformation between two sections was moderate, the same result could often be achieved with a few manual matches. Finally, for each section interface, further manual matches or corrections were applied to achieve the desired quality. An experienced user processed a section interface in approximately 10 minutes. The complete stack could thus be constructed within one hour. 25–35 % of the matches were placed manually. Substantial manual effort was especially necessary close to the section boundaries in x and y .

Mitotic Spindle Anaphase

This stack consists of a full spindle in early anaphase. The chromatids split and move towards the poles. The data set includes the complete spindle with centrosomes. It consists of 25 sections with 1100–4200 MTs per section (Table 1 T0265-42). However, one section in the middle of the stack got damaged during sample preparation, and one half of the spindle could, therefore, not be recorded for this section. The automatic alignment of the damaged section failed, which

also affected the following sections, such that half of the stack got distorted in z -direction. With our tool, it was possible to manually steer the alignment and create a reasonable matching for the uncorrupted data. The stack could be completely constructed. Up to 20 manual landmarks and approximately 100 manual matches were used to compute additional 1100 automatic matched MT ends. The MT distribution for the 3 sections around the damaged section is 4176, 2134, and 3774, where the one in the middle is the damaged section. An experienced user was able to handle the two interfaces above and below the damaged section within one hour.

Mitotic Spindle Metaphase

This stack consists of a spindle in metaphase. The full spindle has formed; the chromosomes are located in the center of the cell and are attached to a subset of MTs, called kinetochore MTs. Similarly to the anaphase spindle, the data set includes the complete spindle with centrosomes. The spindle consists of 23 sections with 900–3900 MTs per section (Table 1 T0265-21). In contrast to the previous two examples, the quality of the sections were good. The automatic algorithms worked quite well. We chose this example mainly to test the performance but also to check if the automatic workflow within the new tool works for the complete spindle. The full reconstruction is visualized in Fig 1C and Fig 8D.

4.2 Performance

4.2.1 Memory Usage

During alignment and matching, the tool requires memory for the two tomograms of the selected interface. The alignment and matching can be performed on an advanced desktop system. For our largest data sets, approximately 10 GB were sufficient. For the creation of the stack in full resolution, approximately $(1 + n) \cdot k$ memory is required, where n is the number of sections and k the size of the largest tomogram. For our use cases, 150 GB was sufficient for the reconstruction of the largest stacks in full resolution.

4.2.2 Rendering Performance

The interactivity of the application mainly depends on the performance of the rendering algorithms and the required internal data structure updates. The most expensive part for the visualization is the matching mode, where three views display both the bottom section and the warped top section with microtubules. Furthermore, a map depicts the end states of the MTs of the bottom section. The rendering performance depends on the resolution of the views. For our tests, Amira was running on a screen with 1920×1200 pixels. The view size of all four views together was 1400×1000 . For small and mid-sized sections, we measured up to 25 frames per second, while for the largest data sets the performance decreased to 12 frames per second on an Intel Xeon i7 X5650 with 6 cores. Thus, with default desktop systems, it was possible to align and match serial sections for all our use cases. The CPU was the bottleneck for the full rendering pipeline. It is used for the detection of the intensity values of the

Table 1: Serial section stacks assembled by our collaborators. Data sets T0208-1, T0391-2, T0475, and T0265-21 are visualized in Fig 8.

Data Set	User	#Sections	Size in GB	#Microtubules
T0275-1	0	13	9.6	1511
T0275-3	0	7	9.2	1132
T0275-5	0	6	9.7	505
T0275-9	0	13	8.0	1289
T0275-10	0	14	8.9	1503
Nocodazole-1	1	17	8.2	248
Nocodazole-2	1	14	6.4	299
SAS4	1	12	11.9	7524
T0391-1	2	6	2.8	246
T0391-2	2	11	10.1	1403
T0391-3	2	14	9.0	1611
T0391-4	2	5	1.9	671
T0391-5	2	8	3.2	893
T0391-6	2	17	16.9	2553
T0391-7	2	25	16.4	2212
T0391-8	2	11	9.5	2418
T0475	3	22	47.6	4884
T0479	3	29	79.8	8051
T0252 [3]	4	24	13.6	3581
T0265-21 [3]	4	23	79.3	27051
T0265-41 [3]	4	15	28.7	11801
T0265-42 [3]	4	25	93.1	12205
T0265 [3]	4	15	36.4	8774
T0345-11	4	7	10.7	1892
T0208-1 [4]	4	20	11.4	3723

slices and for the warping of the slices and microtubules. The rendering of the slices and the ray casting of the MTs and landmarks was negligible.

5 Discussion

An important design decision is that the construction of the complete stack is decomposed into smaller tasks that solve the alignment and stitching problem separately for each interface between two neighboring sections. This seems reasonable based on the assumption that the alignment and matching for one interface is independent from other interfaces. Such a workflow has advantages for both, the visual design as well as the performance.

5.1 Visual Design

Two z -slices of neighboring sections together with a blended interactive overlay preview to align neighboring sections has been successfully used for serial sections images before [11]. Our observations confirm that the approach works well and allows a user to quickly visually assess the alignment.

Due to the low signal-to-noise ratio, especially for electron tomography data, rendering of slices seems superior to volume rendering or other 3D rendering techniques for image data. Slice rendering directly displays the intensity values, is straightforward to implement, and scales reasonably with image data size. We doubt that 3D visualizations of the image data would help to improve the alignment and matching. Moreover, 3D techniques often require more parameter adjustments than slice rendering, such as transfer functions, and may therefore be more time-consuming to control in practice.

A stack can contain thousands of densely arranged MTs. Visualizing the MTs as 3D lines for the full stack is inadequate for inspecting the matching at section interfaces. Due to occlusion, navigation in dense sets of 3D filamentous structures is inherently difficult. However, if the 3D visualization is restricted to a certain location, it can help to match the filaments in difficult setups, because the slope of the filaments can be seen in all directions at once, which can be sometimes difficult in a slice. On the other hand, showing MT ends as overlay of two z -slices allows the user to easily detect patterns in both sections, which can be more difficult in a 3D visualization that allows arbitrary viewing angles. For this reason we decided to provide both, a full 2D view of the filaments within the slice and a local 3D view.

For rendering the MTs, cylinders seem to be superior to thin lines. The intersection of cylinders with slices creates ellipsoids instead of points. The shape of the ellipsoids allows perceiving the slope of the MTs, which helps to detect corresponding ends.

The map that depicts the microtubules end states was inspired by real-time strategy games. Most of these games offer a map that is initially black to indicate undiscovered regions. During the game, the player discovers regions with units. The visited terrain becomes visible on the map; the black is replaced by map details. Games usually distinguish further between regions that are currently under control by the player and regions that were visited before but are currently not under control. While the latter are usually visualized a bit darker, the others show additional information such as enemies in the regions. In our case, the bottom section is initialized as an undiscovered map. Automatic unconfirmed MT ends correspond to ‘visited’ regions. The confirmed MT ends correspond to regions ‘under control’.

5.2 Techniques

The design decision to restrict the workflow to processing interfaces between two neighboring sections one after the other has several advantages. Handling data and rendering of the full stack at once can be avoided. Large-scale image data visualization could be quite complex and often requires advanced out-of-core rendering techniques [14]. Since we restricted the design to a single interface, simple and efficient slice visualizations can be rendered interactively without complex data handling. The most challenging part is the real-time warping, which we accelerated by grid data structures to achieve interactive frame rates (Sec 3.2). We expect that the design would handle increasing tomogram sizes without major modifications. Because most visualizations are based on screen-space, the tool should scale reasonably with data size. The integration of alternative automatic alignment and matching algorithms should be straightforward, since the design does not use any advanced data structures.

As described in Sec 2.2, the top left view in alignment mode is rotated and translated to approximately match the warping applied to the overlay viewer. We tried two approaches for this adjustment before we decided to use the global approach. The transformation could be calculated either by the whole slice (global) or by the current view of the bottom section (local). Although the local option more closely approximates the current local transformation, it is also visually less stable and intuitive to use, because the viewer may adjust rapidly during zooming. A global rigid transformation is more stable and was sufficient in all test cases to display the corresponding regions in both views.

5.3 User Feedback

Our collaborators, some of which are co-authors, have been involved since the early design phases of the tool and influenced several design decisions. For this reason, we decided to only informally discuss user feedback and main lessons learned, instead of reporting user feedback more formally.

Especially, the interactive visual feedback of the whole process that shows the alignment and matching for the section interfaces was important to the users to assess the quality. For the alignment, our collaborators quickly considered our design good enough. For the matching, however, they requested more options and better solutions, which led us to create the stepping tool to confirm MT ends and the visualization of the MT end state map.

Depending on the data set, it was impossible to check all MT ends per section interface. To achieve a good distribution over the whole section interface, we decided to compute a random order of the MT ends for the stepping tool. Another idea was to select an order where the next MT end is always farthest away from all previously confirmed MT ends with periodic boundary conditions. This could potentially generate a distribution that would more quickly cover the entire section. It would, however, require more computation time and seemed irrelevant for our use cases. Although the stepping tool alone already helped validating the matching, our collaborator found it difficult to estimate the overall matching quality for the entire interface. For this reason, we introduced the map that depicts the state of the MT ends to give an impression of the overall progress.

Our collaborators said that they found the lines in the overlay view that indicate position and orientation of the side views very helpful. The lines primarily help in two ways. First, they provide a visual link between the overlay view and the side view. Second, the selected MT end is highlighted by the cross where the two lines intersect and is thus easier to inspect.

Originally, the automatic landmarks were maintained separately from the matching process. The coherent point drift algorithm could be applied completely independently from the matching, as in the original fully automatic pipeline. We observed that this separation was confusing in practice. The landmarks are computed based on the geometry of the MT ends in the matching region and thus conceptually belong to the MT ends. We observed that users either accepted these landmarks or rejected them, because the alignment completely failed. They rarely modified the automatic landmarks. For this reason, we decided to move the original alignment into a hidden pre-processing step before the automatic matching, which is only computed if no automatic landmarks are available (during initialization or when a user decided to delete all

automatic landmarks). We observed that if the alignment fails, the matching also fails, and users rejected both. They then placed manual landmarks in combination with a few manual matches, which improved the next automatic matching computation without the automatic alignment. In combination with this workflow, we decided to always recompute the automatic landmarks after the matching for two main reasons. First, only matched MT ends should create landmarks, because the other MT ends possibly do not belong to each other. Second, during the manual matching, one can directly see the corresponding alignment changes due to the new landmark. This intuitive workflow combines local alignment and matching into a single control handle.

We received positive feedback that the manual and automated matching can be applied iteratively to progress towards a good stitching result. It was possible to create reasonable solutions even for cases where the fully automatic stitching process failed.

6 Conclusion and Future Work

In this paper, we proposed a tool to align serial section tomograms in combination with the stitching of filamentous structures, specifically MTs in our application. The tool allows users to visually validate the stitching process and to support the automatic algorithms by placing manual landmarks and matches. Cases for which the fully automatic workflow failed can now be solved with a reasonable investment of manual labor, because a user can steer the whole process of stitching a serial section stack.

While stitching sections, users often identify minor errors of the tracing. These can be, for example, false positive MTs or split MTs that should be connected. Currently, it is not possible to edit the MT geometry with our tool. Users either need to improve the MT tracing before or after the stitching. It would be helpful to add such a functionality; it would allow users to immediately correct errors in the MT segmentation. For example operations would be possible to add, split, remove, or merge MTs.

An open question is the computation or manual estimation of the gap between two adjacent sections. Currently, we expect that the user crops the tomograms in z such that they fit seamlessly. It could be helpful to support setting the cropping regions interactively for each interface in the tool. This would probably improve the selection of cropping parameters and could, thus, also improve the alignment and matching procedure. It might be possible to compute good cropping parameters automatically from confirmed matches.

Acknowledgments

This work was funded by the German Research Foundation DFG Grants MU 1423/8-1 and 10-1 to T.M.-R. and DFG Grant PR 1226/4-1 to S.P. R.K. received funding from the European Union’s Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 675737 (grant to T.M.-R.). We would like to thank E. Szentgyoergyi and M. Kirchner for providing different spindle data sets and for their comprehensive user feedback. We would like to thank D. Baum for helping prepare the software release.

Software

The software is available at <https://www.zib.de/software/serial-section-aligner>.

References

- [1] O’Toole E, van der Heide P, Richard McIntosh J, Mastronarde D. In: Hanssen E, editor. *Large-Scale Electron Tomography of Cells Using SerialEM and IMOD*. Cham: Springer International Publishing; 2018. p. 95–116.
- [2] Müller-Reichert T, Srayko M, Hyman A, O’Toole ET, McDonald K. Correlative Light and Electron Microscopy of Early *Caenorhabditis elegans* Embryos in Mitosis. In: *Cellular Electron Microscopy*. vol. 79 of *Methods in Cell Biology*. Academic Press; 2007. p. 101–119.
- [3] Redemann S, Baumgart J, Lindow N, Shelley M, Nazockdast E, Kratz A, et al. *C. elegans* Chromosomes Connect to Centrosomes by Anchoring into the Spindle Network. *Nature Communications*. 2017;8:15288. doi:10.1038/ncomms15288.
- [4] Redemann S, Lantzsch I, Lindow N, Prohaska S, Srayko M, Müller-Reichert T. A Switch in Microtubule Orientation During *C. elegans* Meiosis. *Current Biology*. 2018;28:2991–2997.e2. doi:10.1016/j.cub.2018.07.012.
- [5] Redemann S, Weber B, Möller M, Verbavatz JM, Hyman AA, Baum D, et al. The Segmentation of Microtubules in Electron Tomograms Using Amira. In: *Mitosis: Methods and Protocols*; 2014. p. 261–278.
- [6] Weber B, Greenan G, Prohaska S, Baum D, Hege HC, Müller-Reichert T, et al. Automated Tracing of Microtubules in Electron Tomograms of Plastic Embedded Samples of *Caenorhabditis elegans* Embryos. *Journal of Structural Biology*. 2012;178(2):129–138. doi:10.1016/j.jsb.2011.12.004.
- [7] Weber B, Tranfield EM, Höög JL, Baum D, Antony C, Hyman T, et al. Automated Stitching of Microtubule Centerlines across Serial Electron Tomograms. *PLOS ONE*. 2014;9(12):e113222. doi:10.1371/journal.pone.0113222.
- [8] Tasdizen T, Koshevoy P, Grimm BC, Anderson JR, Jones BW, Watt CB, et al. Automatic Mosaicking and Volume Assembly for High-Throughput Serial-Section Transmission Electron Microscopy. *Journal of Neuroscience Methods*. 2010;193(1):132–144. doi:10.1016/j.jneumeth.2010.08.001.
- [9] Saalfeld S, Cardona A, Hartenstein V, Tomančák P. As-Rigid-as-Possible Mosaicking and Serial Section Registration of Large ssTEM Datasets. *Bioinformatics*. 2010;26(12):i57–i63.
- [10] Kremer JR, Mastronarde DN, McIntosh JR. Computer Visualization of Three-Dimensional Image Data Using IMOD. *Journal of Structural Biology*. 1996;116(1):71–76. doi:10.1006/jsbi.1996.0013.

- [11] Anderson JR, Jones BW, Yang JH, Shaw MV, Watt CB, Koshevoy P, et al. A Computational Framework for Ultrastructural Mapping of Neural Circuitry. *PLOS Biol.* 2009;7(3):1–20. doi:10.1371/journal.pbio.1000074.
- [12] Berlanga ML, Phan S, Bushong EA, Wu S, Kwon O, Phung BS, et al. Three-dimensional Reconstruction of Serial Mouse Brain Sections: Solution for Flattening High-Resolution Large-Scale Mosaics. *Frontiers in Neuroanatomy.* 2011;5(17). doi:10.3389/fnana.2011.00017.
- [13] Cardona A, Saalfeld S, Schindelin J, Arganda-Carreras I, Preibisch S, Longair M, et al. TrakEM2 Software for Neural Circuit Reconstruction. *PLOS ONE.* 2012;7(6):1–8. doi:10.1371/journal.pone.0038011.
- [14] Beyer J, Hadwiger M, Al-Awami A, Jeong WK, Kasthuri N, Lichtman JW, et al. Exploring the Connectome: Petascale Volume Visualization of Microscopy Data Streams. *IEEE Computer Graphics and Applications.* 2013;33(4):50–61. doi:10.1109/MCG.2013.55.
- [15] Ai-Awami AK, Beyer J, Haehn D, Kasthuri N, Lichtman JW, Pfister H, et al. NeuroBlocks - Visual Tracking of Segmentation and Proofreading for Large Connectomics Projects. *IEEE Trans Vis Comput Graph.* 2016;22(1):738–746. doi:10.1109/TVCG.2015.2467441.
- [16] Dercksen VJ, Hege HC, Oberlaender M. The Filament Editor: An Interactive Software Environment for Visualization, Proof-Editing and Analysis of 3D Neuron Morphology. *NeuroInformatics.* 2014;12(2):325–339. doi:10.1007/s12021-013-9213-2.
- [17] Stalling D, Westerhoff M, Hege HC. Amira: a Highly Interactive System for Visual Data Analysis. In: *The Visualization Handbook.* Elsevier; 2005. p. 749–767.
- [18] Harrower M, Brewer CA. ColorBrewer.org: an Online Tool for Selecting Colour Schemes for Maps. *Cartographic Journal, The.* 2003;40(1):27–37.
- [19] Schaefer S, McPhail T, Warren J. Image Deformation Using Moving Least Squares. *ACM Trans Graph.* 2006;25(3):533–540. doi:10.1145/1141911.1141920.
- [20] Sigg C, Weyrich T, Botsch M, Gross M. GPU-Based Ray-Casting of Quadratic Surfaces. In: Botsch M, Chen B, Pauly M, Zwicker M, editors. *Symposium on Point-Based Graphics.* The Eurographics Association; 2006.

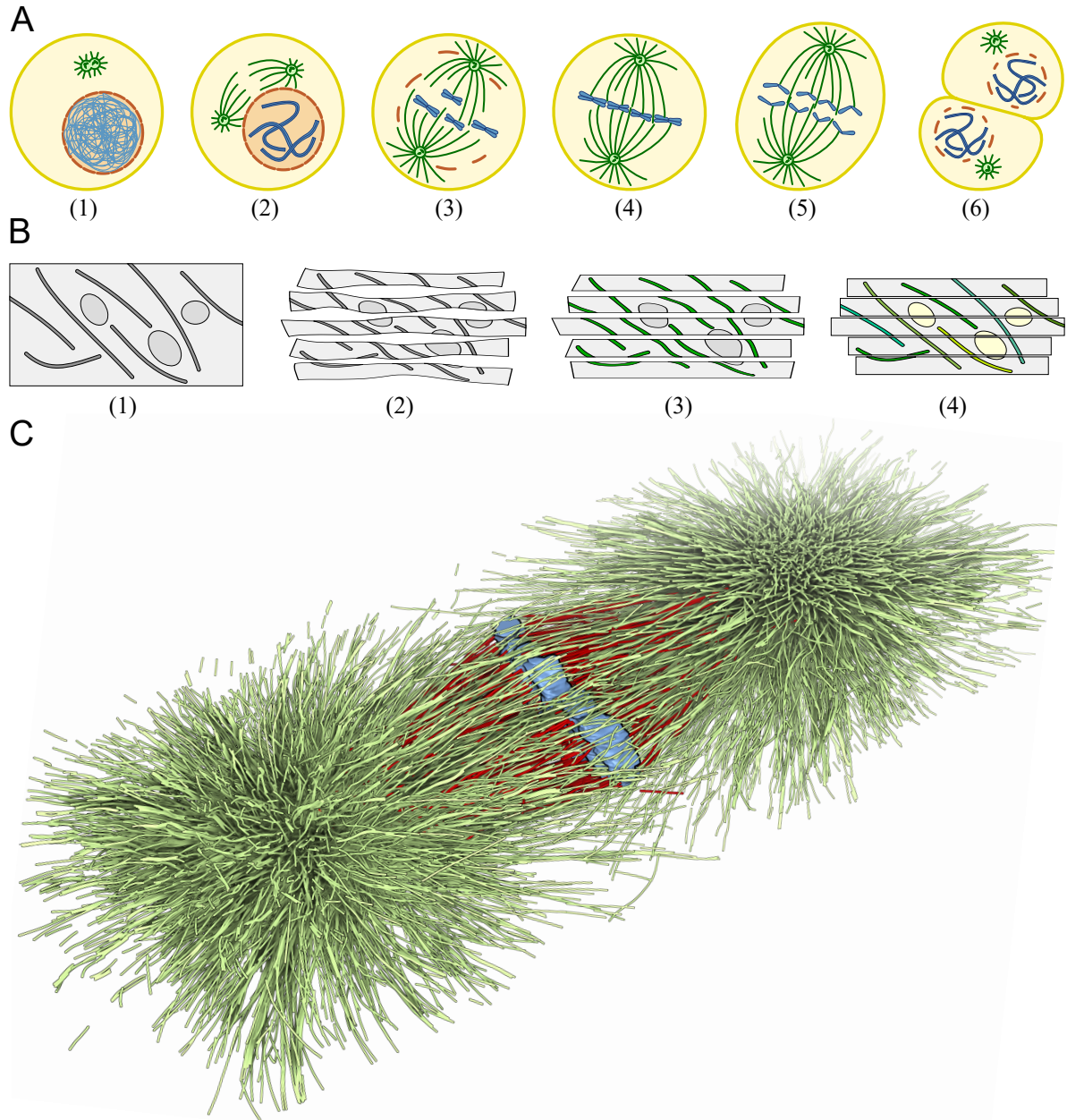
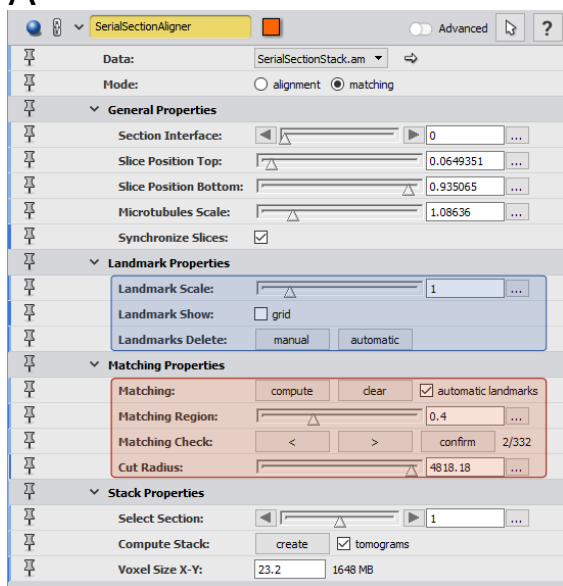


Figure 1: Overview of mitotic cell division, schematic of the reconstruction pipeline, and 3D representation of the final spindle reconstruction. (A) Illustration of the main phases of cell division: Interphase (1), prophase (2), prometaphase (3), metaphase (4), anaphase (5), and telophase (6) with subsequent cytokinesis. During mitosis, a bipolar spindle is built from microtubules (represented as green lines). (B) Reconstruction pipeline: A specimen (1) is physically cut and imaged, which leads to a serial stack of distorted sections (2). Filaments are segmented within the sections (3). Finally, the sections are stitched and the filaments matched (4). (C) Completely aligned and stitched mitotic metaphase spindle in the early embryo of the nematode *Caenorhabditis elegans* with approximately 27,000 microtubules (shown in green and red), Table 1 T0265-21. The spindle was reconstructed from 23 serial section tomograms. Chromosomes are depicted in blue. Microtubules attached to the chromosomes are shown in red.

A



B

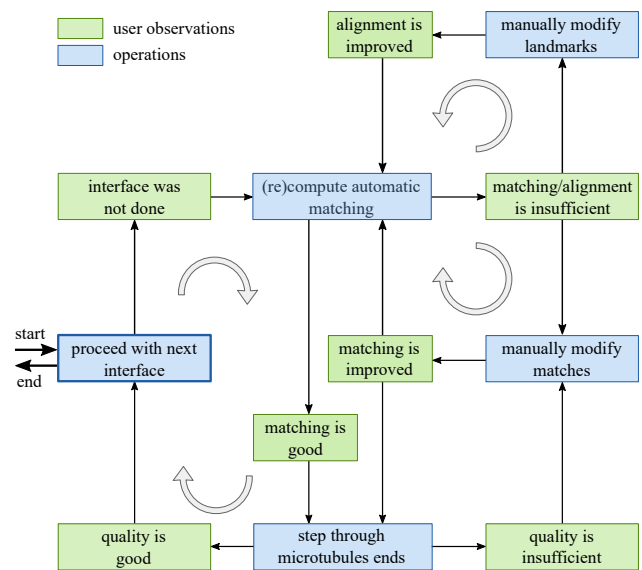


Figure 2: **Graphical user interface and recommended workflow.** (A) Graphical user interface of the aligner module. The blue part is only visible for the alignment mode and the red part only for the matching mode. (B) Recommended workflow for the stitching process. Observation tasks (green boxes) lead to operations (blue boxes). The most frequently used cycles are highlighted by the grey circular arrows.

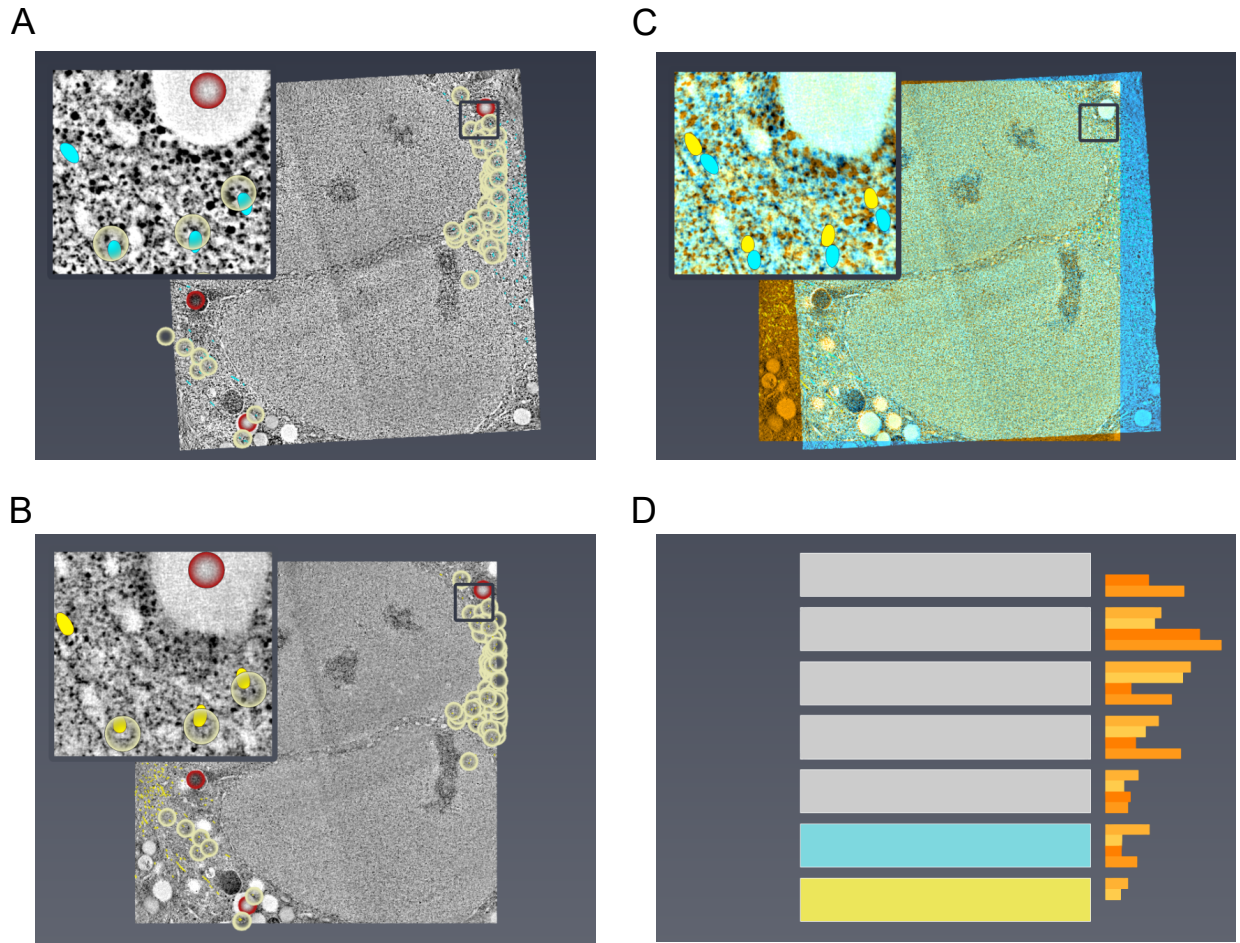


Figure 3: **The four views of the alignment mode.** Two views show z-slices of the top section (A) and the bottom section (B) with a warping preview as overlay view (C) as well as an abstract visualization of the full stack (D) with the selected interface and a vertical histogram of the end point density.

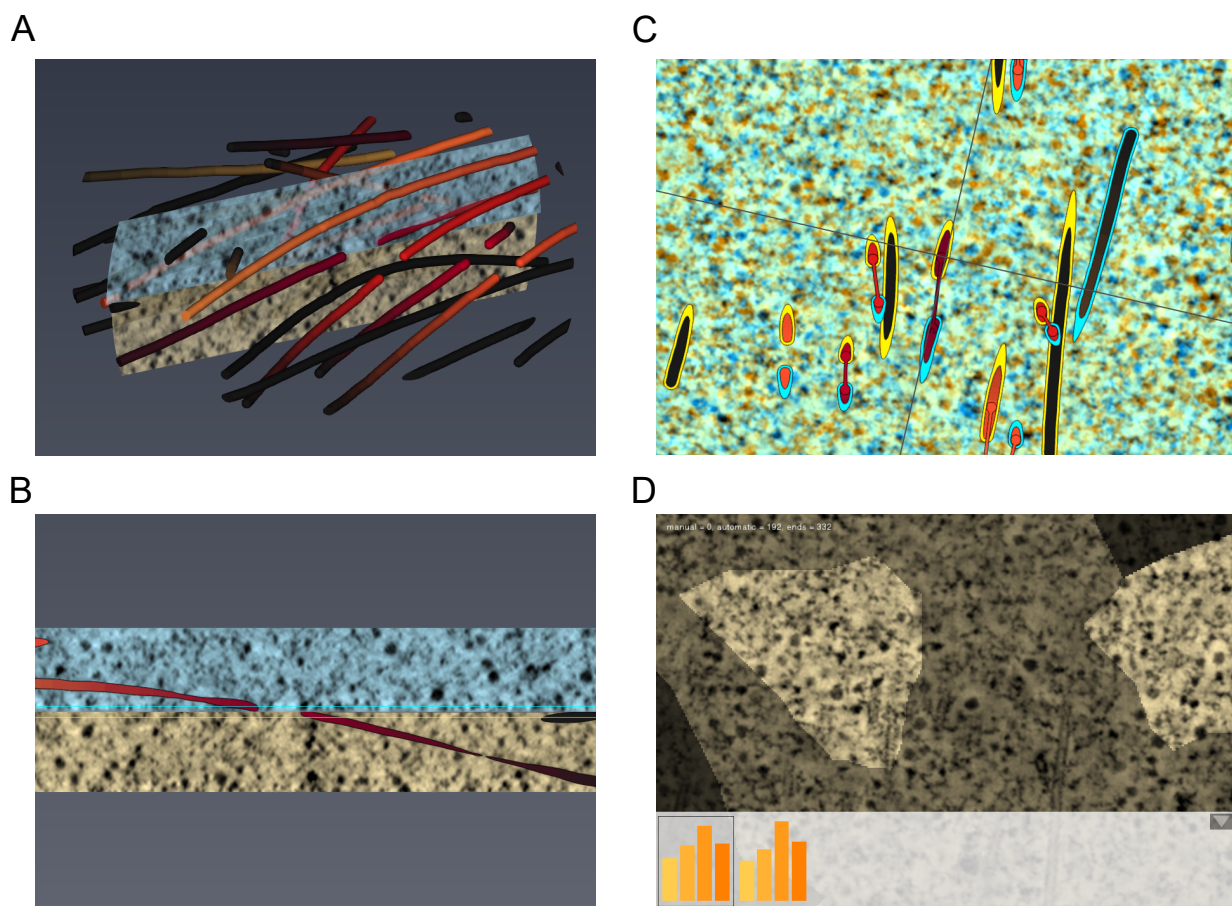


Figure 4: **The four views of the matching mode.** (A) Local 3D view. (B) Orthogonal side view. (C) Warped overlay of the bottom and the top section. (D) Matching state map of the MT ends of the bottom section.

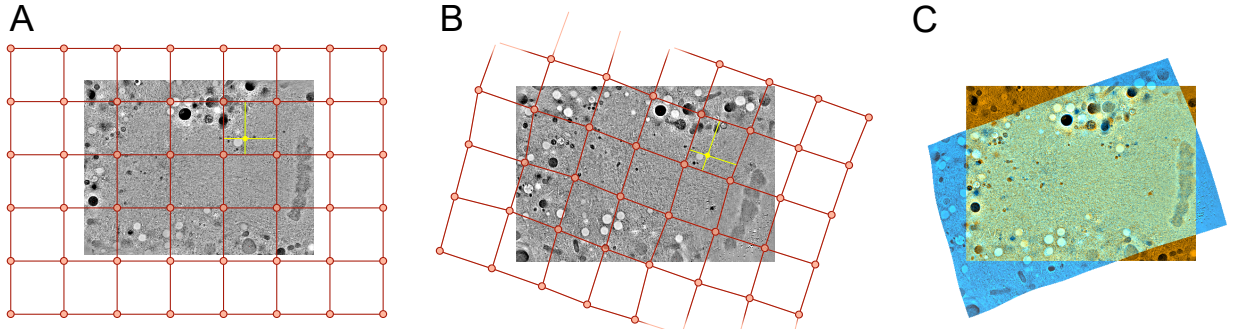


Figure 5: **Warping of the z -slice of the top section.** (A) A 2D grid in the space of the bottom section is generated that fills the complete view. (B) The grid is transformed to the space of the top section. (C) The transformation of all visible points of the bottom section is approximated by a bilinear interpolation in the transformed grid. Then the intensity value is captured and blend with the bottom section.

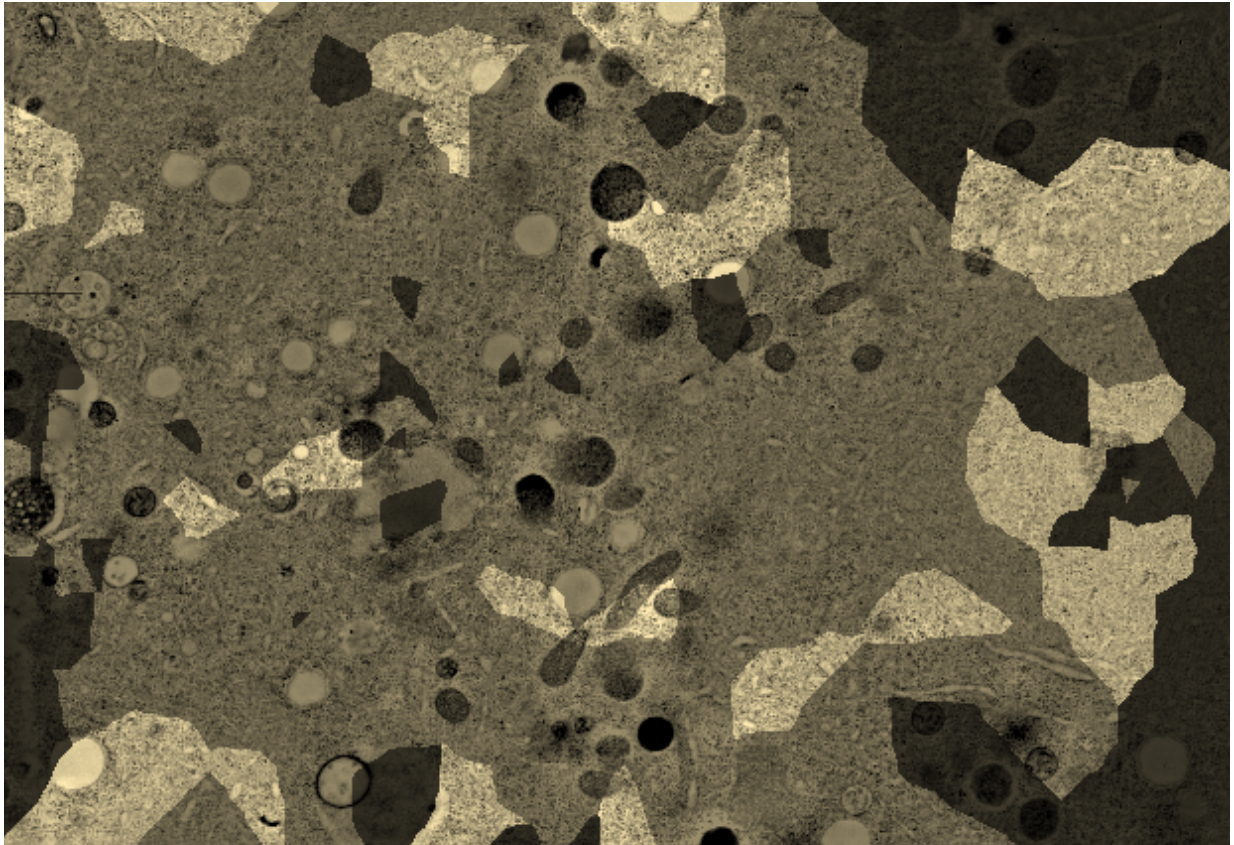


Figure 6: **Filament end state map.** Dark regions correspond to unmatched unconfirmed filament ends, bright regions to confirmed ends, and the ends with medium brightness are automatically matched but not confirmed.

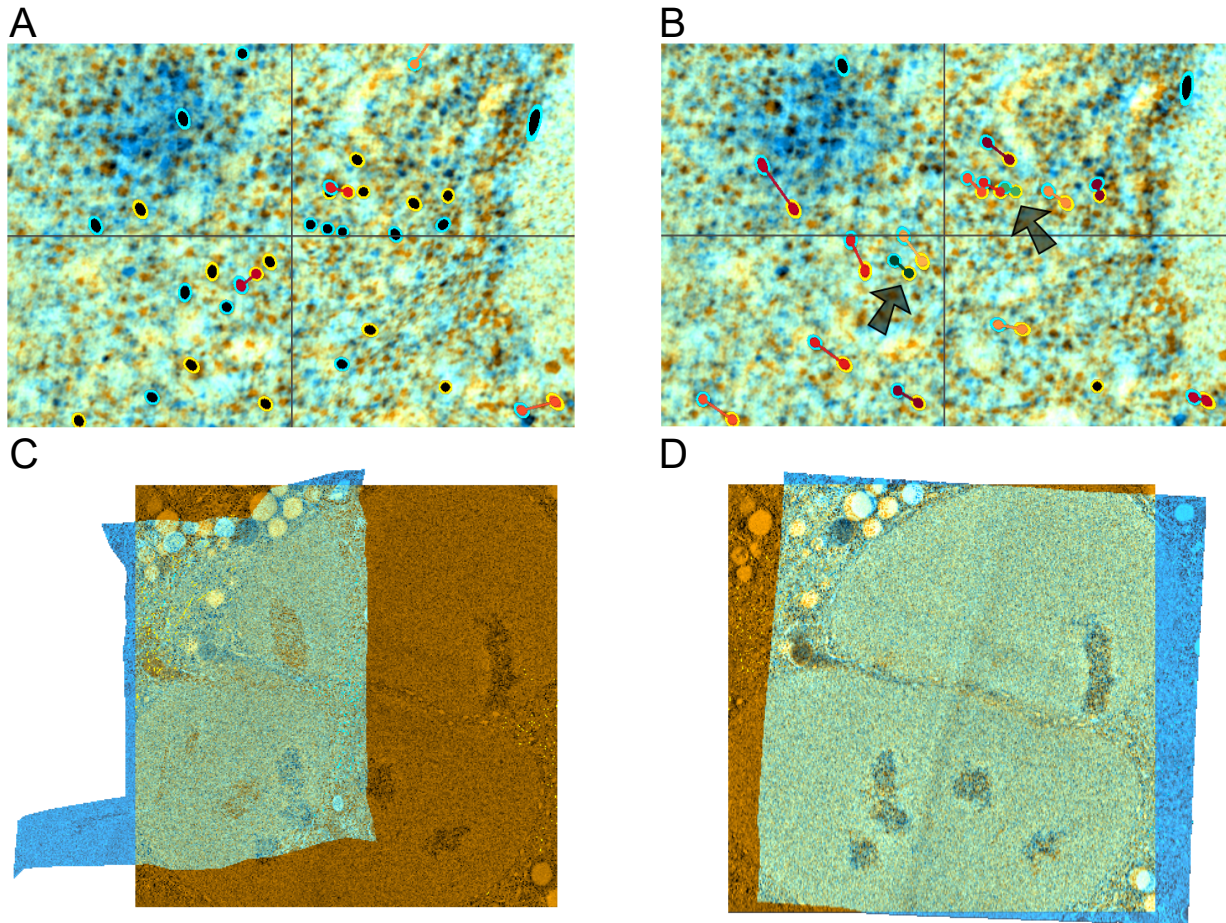


Figure 7: **Fully automatic approach vs. semi-automatic workflow.** (A) The automatic matching failed in this region, but one can clearly see the matching patterns, Table 1 T0265-42. (B) By adding two manual matches (in green) the automatic matching for the complete region worked. (C) Alignment result of the original automatic pipeline in a *C. elegans* prometaphase data set of one section interface, Table 1 T0345-11. (D) Solution of the problem by adding a few manual filament matches.

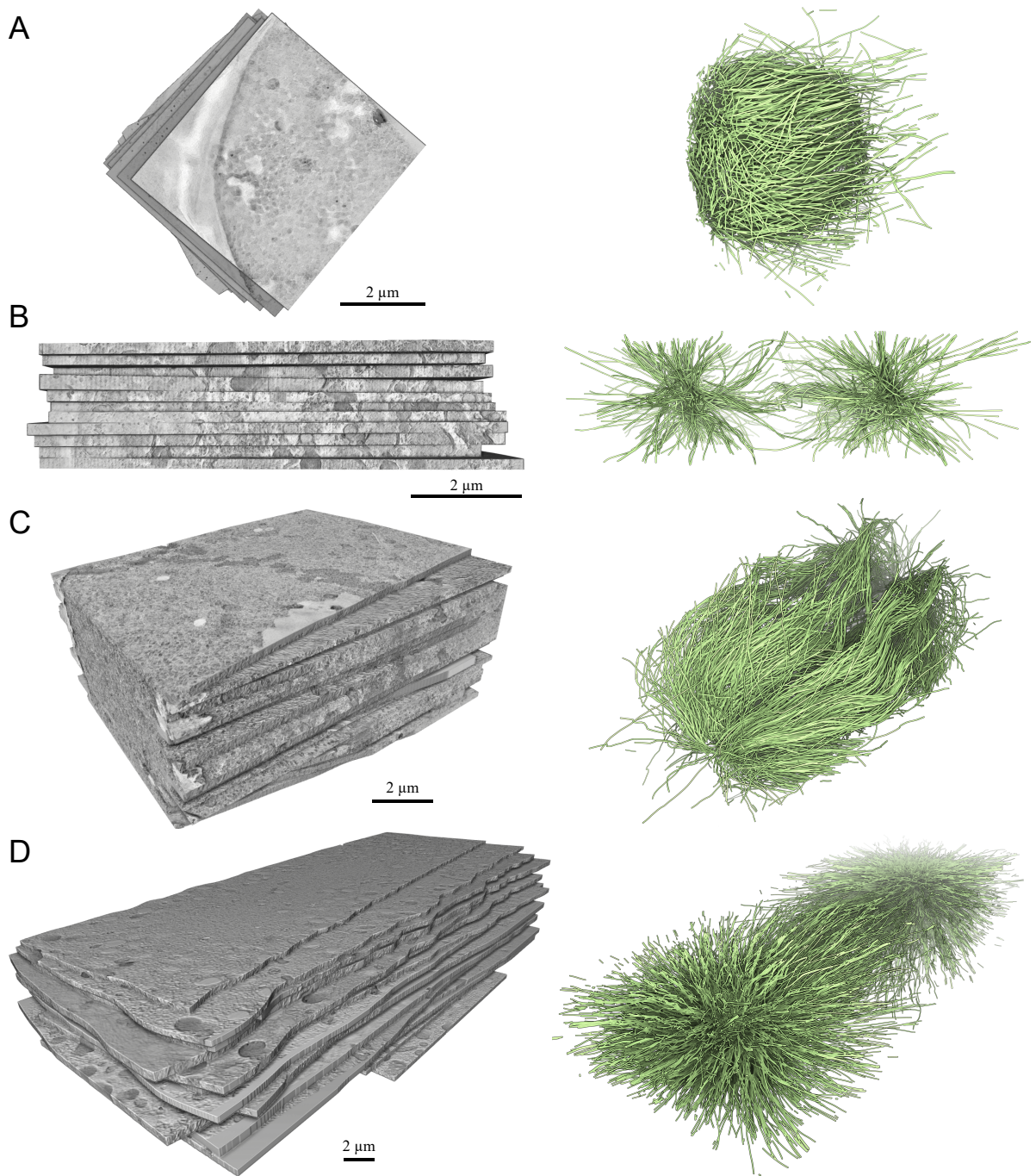


Figure 8: **Three-dimensional reconstruction of microtubule-based spindles of different biological samples.** Tomographic stacks are shown on the left, 3D reconstructions of the spindles on the right. (A) Anaphase I of female meiosis in *C. elegans*, Table 1 T0208-1. (B) Anaphase I of male meiosis in *C. elegans*, Table 1 T0391-2. (C) Mitotic metaphase in a mammalian cell line (HeLa cells), Table 1 T0475. (D) First mitotic metaphase in the early *C. elegans* embryo, Table 1 T0265-21.