Semi-automatic stitching of filamentous structures in image stacks from serial-section electron tomography

Norbert Lindow1  |  Florian N. Brünig1  |  Vincent J. Dercksen1  |  Gunar Fabig2  |  Robert Kiewisz2  |  Stefanie Redemann3,4,5  |  Thomas Müller-Reichert2  |  Steffen Prohaska1  |  Daniel Baum1

1 Department of Visual and Data-Centric Computing, Zuse Institute Berlin, Berlin, Germany
2 Experimental Center, Faculty of Medicine Carl Gustav Carus, Technische Universität Dresden, Dresden, Germany
3 School of Medicine, Center for Membrane and Cell Physiology, University of Virginia, Charlottesville, Virginia
4 School of Medicine, Department of Molecular Physiology and Biological Physics, University of Virginia, Charlottesville, Virginia
5 School of Medicine, Department of Cell Biology, University of Virginia, Charlottesville, Virginia

Abstract: We present a software-assisted workflow for the alignment and matching of filamentous structures across a three-dimensional (3D) stack of serial images. This is achieved by combining automatic methods, visual validation, and interactive correction. After the computation of an initial automatic matching, the user can continuously improve the result by interactively correcting landmarks or matches of filaments. Supported by a visual quality assessment of regions that have been already inspected, this allows a trade-off between quality and manual labour. The software tool was developed in an interdisciplinary collaboration between computer scientists and cell biologists to investigate cell division by quantitative 3D analysis of microtubules (MTs) in both mitotic and meiotic spindles. For this, each spindle is cut into a series of semi-thick physical sections, of which electron tomograms are acquired. The serial tomograms are then stitched and non-rigidly aligned to allow tracing and connecting of MTs across tomogram boundaries. In practice, automatic stitching alone provides only an incomplete solution, because large physical distortions and a low signal-to-noise ratio often cause experimental difficulties. To derive 3D models of spindles despite dealing with imperfect data related to sample preparation and subsequent data collection, semi-automatic validation and correction is required to remove stitching mistakes. However, due to the large number of MTs in spindles (up to 30k) and their resulting dense spatial arrangement, a naive inspection of each MT is too time-consuming. Furthermore, an interactive visualisation of the full image stack is hampered by the size of the data (up to 100 GB). Here, we present a specialised, interactive, semi-automatic solution that considers all requirements for large-scale stitching of filamentous structures in serial-section image stacks. To the best of our knowledge, it is the only currently available tool which is able to process data of the type and size presented here. The key to our solution is a careful design of the visualisation and interaction tools for each
processing step to guarantee real-time response, and an optimised workflow that efficiently guides the user through datasets. The final solution presented here is the result of an iterative process with tight feedback loops between the involved computer scientists and cell biologists.

**KEYWORDS**
Amira, automatic segmentation, electron tomography, filamentous structures, microtubules, serial sectioning, software, stitching

1 | INTRODUCTION

The segregation of chromosomes is an essential biological process and indispensable for each dividing cell. Failures in segregation 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 mechanisms of chromosome segregation is of significant biological importance.

In eukaryotic cells, chromosome segregation occurring during both mitosis and meiosis is achieved by bipolar microtubule (MT)-based spindles. The different stages of mitotic cell division are illustrated in Figure 1A. During prophase, the centrosomes start to move towards opposite sides of the cell, and a bipolar spindle mainly consisting of MTs and other associated proteins begins to form. At this point, the chromosomes begin to condense and become visible. The chromosomes then move towards the centre of the cell at prometaphase. In metaphase, chromosomes are aligned at the metaphase plate, and sister chromatids are segregated subsequently in anaphase. Finally, the mitotic spindle is disassembled, and the nuclear membrane around the chromosomes is re-built in telophase. Quantitative analysis of MT arrangement is crucial for an understanding of spindle assembly and organisation.

To study the architecture of spindles with single-microtubule resolution, three-dimensional (3D) reconstruction using electron tomography is an excellent tool. When combined with appropriate techniques for specimen preparation, electron tomography allows an ultrastructural investigation of spindles at different stages of cell division and also a comparison of wild-type and mutant spindles, such as in the model system *Caenorhabditis elegans*.

In electron tomography of biological samples, imaging of serial sections is applied to reconstruct structures in 3D that are too large to image them directly (Figure 1B). For this, samples are cryo-immobilised, embedded in plastic (Figure 1B(a)), and sectioned. It is important to point out here that the quality of sample preparation is of crucial importance for 3D reconstruction as samples of insufficient ultrastructural preservation can never give high-quality representations of the biological sample. The serial sections are then imaged individually, generating a tilt series from which one tomogram is reconstructed for each section (Figure 1B(b)). The tomograms then need to be stitched together along the z-direction to fully reconstruct the recorded cellular object (Figure 1B(c, d)). To expand the imaged x/y area, each serial section can consist of separate tiles that are acquired separately and then merged into a larger tomogram. Typically, such reconstructions from tiles and sections are done in ‘reverse order’, that is, first, tiles are aligned to reconstruct a single section, called ‘montaging’, and second, the serial sections are combined to a complete stack. Since the montaging is well established, in this paper, we focus on the second step and assume that the first step was correctly processed (ie in a sense that the montaging of the individual tomograms per section has been carried out properly and without any misalignment). Depending on possible deformations caused by the physical sectioning and the interaction of the specimen with the electron beam during imaging, this stitching often requires a non-linear alignment. Corresponding structures in neighbouring sections can be identified either based purely on image data or on segmented features.

Typically, the spindle geometry is reconstructed in two steps (Figure 1B(c, d)). In the first step, the MTs are segmented in each serial tomogram separately using template matching. This is followed by application of an automatic tracing algorithm in combination with data verification by users. In the second step, the segmented MTs are used to automatically stitch the serial tomograms to a single stack. Due to the local deformations during imaging, a non-linear alignment is required that often cannot be immediately determined due to the lack of obvious image features. Thus, tomograms have to be aligned by applying a variant of the coherent point drift algorithm (CPD) to the segmented MTs. For each interface between two serial tomograms, the algorithm uses the MT ends of the two neighbouring tomograms and the 3D directions at the MT ends and calculates two-dimensional (2D) landmarks.
FIGURE 1  Cell division and three-dimensional spindle reconstruction. (A) Schematic illustration of mitotic stages: (a) prophase, (b) prometaphase, (c) metaphase, (d) anaphase and (e) telophase with subsequent cytokinesis. During mitosis in animal cells, microtubules (green lines) connect the centrosomes (green circles) and the chromosomes (blue). The nuclear envelope (orange) is usually disassembled during mitosis. (B) Pipeline for the reconstruction of a mitotic spindle: (a) specimen with microtubules (lines) and other cellular components (circles), (b) physical cutting and imaging of the specimen leading to a stack of distorted serial sections, (c) segmentation of cellular features within the flattened sections and (d) stitching of tomograms to match filaments. (C) Completely aligned and stitched mitotic spindle at metaphase in the early embryo of the nematode C. elegans with approximately 27,000 microtubules shown in green and red. The spindle was reconstructed from 23 serial tomograms (Table 1, T0265-21). Chromosomes are shown in blue, kinetochore microtubules in red.

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 tomogram boundaries. With this approach, we previously achieved a 3D reconstruction of the first mitotic spindle in C. elegans at a resolution of 5-8 nm, with up to 5 GB for each of the 30 tomograms (Figure 1C). To the best of our knowledge, no other tool currently exists, with which a similar reconstruction could be achieved.

In our previous work, stitching was designed and implemented as a fully automatic pipeline, which allowed only a final verification after completion of the stack, thus only small manual corrections were realistic for the fully assembled stack. As an example, when the alignment completely failed for one section interface, it was
impossible to create a meaningful stack. We, therefore, developed the semi-automatic solution as presented in this paper that also allows the reconstruction of serial-section stacks with imperfect and thus difficult data. Although this semi-automatic solution was 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 across tomograms, which seems unrealistic with generic image-based registration approaches such as the one used in IMOD.6

A fully automatic approach to stitch such tomograms is often problematic for two main reasons. First, the data quality may be poor due to inaccuracies during specimen preparation and image acquisition, as sections can suffer from physical damage during microscopy. As a result, sections may have been deformed or ruptured, causing the alignment to fail. Second, visual validation and correction of the stitching result is difficult. An interactive inspection of each filament requires a prohibitive amount of time. With limited time, it is not obvious which subset to choose for inspection. There is also no procedure to interactively correct the filament alignment and 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 tomograms, often the full stack could not 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 general, automatic procedures 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.5 It is used in many laboratories for semi-automatic montaging and flattening of tomograms. A tool focusing on the alignment of serial tomograms of electron microscopy images was proposed by Anderson et al.7 This 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 tomograms containing filamentous structures. In contrast to thin plate splines, however, we use moving least squares for the warping. Furthermore, in addition to manually placed landmarks, we use automatically computed landmarks for the final alignment. Berlanga et al proposed a workflow for reconstructing 3D images from serial-section confocal microscopy for the mouse brain.8 Their workflow includes the flattening of the tomograms using manual landmark generation with IMOD as well as semi-automatic alignment of the section with ir-tweak. However, it does not allow one to incorporate automatically computed landmarks in the alignment process. In 2012, Cardona et al reported on a tool for investigating neural circuits from serial-section microscopy images.9 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 proposed a framework that enables petascale volume visualisation.10 The complete montaging and sectioning are 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 organise and inspect the whole segmentation process.11 However, the data sets they deal with do not require the alignment and stitching of tomograms. A tool for a related application, called Filament Editor, was proposed by Dercksen et al.12 It allows reconstruction of 3D neuronal networks based on serial optical microscopy sections. The tool provides 3D visualisation of the networks and image volumes as well as 2D slices and allows users to manually edit the networks. The serial tomogram alignment is computed automatically with a linear transformation. No interaction is possible to help the alignment in case of failure. Furthermore, no non-linear transformations are supported. The networks that can be handled are sparse and small compared to the microtubules data sets that are presented here.

Our goal was to stitch microtubules across a whole stack of serial-section electron tomograms in a semi-automatic way allowing for corrections during the stitching process. A number of requirements were identified for such an approach. The user should be able to influence all steps of the semi-automatic 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 reconstruction. The whole pipeline should be supported by visualisations to verify the current state and offer options to interactively correct errors. Finally, the application should be able to interactively handle all
relevant data sizes, which could be up to 30 tomograms with 5 GB each.

In this paper, we present a software-assisted workflow to address all these requirements. Two related problems need to be solved: (1) the alignment problem, where the serial tomograms are transformed such that they fit together and (2) the matching problem, where the filament structures are combinatorially connected between neighbouring tomograms. In summary, we make the following key contributions:

- Design of a user-controlled workflow (Figure 2) 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.
- Implementation of a new visualisation method to monitor tomogram regions that have already been visually inspected by the user.
- Development of the Serial Section Aligner, a new software tool with multi-view visualisations and interactions designed to support this workflow. Thus, users can inspect both the images and the segmented filaments in order to validate and correct the alignment and matching.

The workflow is demonstrated for serial-section electron tomography of both mitotic and meiotic spindles of high-pressure frozen and plastic-embedded material. To the best of our knowledge, our tool is the only currently available one that can deal with a variety of data sets of very different sizes and quality as presented in this paper.

2 | DESIGN AND IMPLEMENTATION

Based on the requirements stated in the previous section, we decided that both alignment and matching steps should be first solved locally for each section interface, that is for the cut between two neighbouring tomograms. Subsequently, the full stack should be constructed from the local solutions. These conclusions lead to the following design decisions for stitching of a single section interface:

- The two tomograms of the interface should be interactively visualised with the current local alignment and matching such that errors can be quickly detected.
- Simple interaction tools should allow one to set or change the alignment with manual landmarks. Additionally, completely discarding the initial automatic matching should be possible.
- Manual landmarks should be used to improve the initialisation of the automatic alignment computation.
- The current matching should be changed 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 our previous fully automatic pipeline, only matched filament endpoints should impact the tomogram alignment in order to control the consistent microtubule geometry reconstruction across tomograms.
- The application should give immediate visual feedback about the incremental matching progress.

The stitching tool was implemented as two modules in the visualisation software Amira. The first module is a data class (Serial Section Stack) that manages the input data and the stitching state. The second module (Serial Section Aligner) implements the visualisations and editing operations of the stitching workflow.

### 2.1 Serial section stack

A Serial Section Stack contains the data regarding all serial tomograms of the stack including the reconstructed filaments. Assume the serial-section stack contains \( n \) tomograms, they will be denoted by \( T_1, \ldots, T_n \). For the stitching of the stack (Figure 1B), we expect that each tomogram has been pre-processed, that is distortions along the thin tomogram side, usually the \( z \)-direction, have been removed in a flattening pre-processing step using IMOD, followed by a cropping step. Furthermore, the filaments have been segmented and verified for each tomogram as described by Weber et al. We will denote the set of filaments of tomogram \( T_i \) by \( F_i \), where each filament \( f \in F_i \) is a piecewise linear curve with \( f = \{ p_1, \ldots, p_k \} \) and \( p_j \in \mathbb{R}^3 \). We further assume that the stack is always built in \( z \)-direction. We call a slice of the tomogram that lies in the \( x-y \)-plane a \( z \)-slice with respect to a certain \( z \)-value.

In addition to the tomograms and their corresponding filament sets, the Serial Section Stack also stores the stitching state for each of the \( n - 1 \) interfaces between neighbouring tomograms. A stitching state contains sets of landmarks on the tomogram boundaries and sets of matchings representing corresponding filament ends.

### 2.2 Serial section aligner

The Serial Section Aligner module provides two modes. The alignment mode allows the user to manually add, modify, and remove landmarks. The matching mode, on the other hand, enables the automatic computation of a matching in combination with manual editing and verification. In order to visualise the current state of the matching and alignment and to allow for manual interaction, the module uses a four-view visualisation (Figure 3). In addition, the module has a graphical user interface (GUI) to adjust the module parameters (Figure 2A). The first part of the GUI is independent of the specific mode. It allows selection of the currently investigated section interface, adjusting visualisation parameters for the interface, changing the position of the two \( z \)-slices of the neighbouring tomograms, respectively, and adjusting the scale of the filament radii. With the advanced module options, 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 active 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. The software is available as extension package for Amira. The application is demonstrated in the supplementary Video S1, which shows how the software tool described in this publication can be applied to stitch filamentous structures in image stacks from serial-section electron tomography. Note that the video does not describe the workflow but the individual operations that can be combined in different ways to form a workflow. Our recommended workflow, shown in Figure 2B, is described and discussed in Section 2.3. The video is subdivided into the following parts (times are given in parentheses): (1) Stack creation (00:07-00:29); (2) Navigation (00:30-01:46); (3) Alignment (01:47-02:46); (4) Matching (02:47-04:05); (5) Check matching (04:06-05:26) and (6) Build stack (05:27-07:02).

#### 2.2.1 Alignment mode

This mode is for adjusting the alignment of a pair of tomograms for a selected interface. The four views in the visualisation window (Figure 3) are used in the following way. The top left view (Figure 3A) displays a \( z \)-slice of the upper tomogram of the selected interface. Accordingly, the bottom left view (Figure 3B) 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 (Figure 2A). The two sliders are synchronised per default; when the top \( z \)-slice is decreased by the user, the bottom one is automatically increased and the other way around. The effect of changing the \( z \)-slice positions is demonstrated in the Video (1:12-1:28). The left views allow the user to identify corresponding features in the tomograms, such as vesicles, chromosomes, or membranes. To explicitly match features in the two \( z \)-slices, the user can add, move or delete manual landmarks with mouse clicks in the two left views. Landmarks always occur as a pair of 2D points, one point for each tomogram. The points are placed such that they mark corresponding biological structures. In our
FIGURE 3 The four views of the alignment mode. (A) Z-slice of the top section. (B) Z-slice of the bottom section. (C) Warping preview of z-slices of top and bottom sections as overlay view. (D) Abstract visualisation of the full stack showing a vertical histogram of the filament endpoint density for each section, where the density is computed as the number of (unmatched) filament ends divided by the volume (25% of each tomogram). The different shades of yellow help distinguish the histogram bins. The selected section interface is highlighted. In all views, blue is used for the top section, yellow for the bottom one. Red circles represent manual landmarks, yellow circles automatic matchings visualisations, they are rendered as red circles on the z-slices. The radii of the circles can be adjusted by the user. During landmark placement, a rigid transformation is computed based on the current landmarks. The transformation globally rotates and translates the top slice, thereby ignoring distortions, such that corresponding regions are displayed for the top and bottom tomogram during navigation (panning and zooming).

In the top right view (Figure 3C), an overlay of the selected z-slices is visualised by using transparency in combination with two different colours: blue for the top slice and yellow for the bottom slice. Based on the interactions in the left views, the z-slice of the upper tomogram is warped in real time such that landmarks of 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 visualising the tomogram data and the landmarks, the filaments that intersect the z-slices are also visualised in all three views. Filaments of the bottom tomogram are displayed in yellow and the filaments of the top tomogram in blue, corresponding to the colours of the tomogram overlay visualisation in the top right view.

Finally, the bottom right view (Figure 3D) 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 colours 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 extent in z-direction, that is the number of unmatched filament ends divided by 25% of each tomogram volume. Here, only the x-y overlap to the nearest neighbour tomogram is considered. The different shades of yellow of the histogram bins help to distinguish the bins and to identify their location within the tomogram. 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
FIGURE 4  The four views of the matching mode. (A) Local 3D view around a matched filament endpoint pair. (B) Orthogonal side view of the matched filament endpoint pair. (C) Warped overlay of the bottom and the top section. Automatically matched endpoint pairs are depicted in shades from bright orange to red, while manually matched pairs are coloured in shades of green. (D) Matching state map of the microtubule ends of the bottom section

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 tomogram boundary compared to the ends within the tomogram indicates an incomplete matching.

In addition to manual landmarks, automatic landmarks (yellow) can be generated from matched filaments. To do so, the ‘automatic landmarks’ checkbox in the GUI (Figure 2A) needs to be selected. This approach is described in the following section. The user can also remove and manipulate these landmarks. However, if the automatic landmark placement is activated, these landmarks will be updated by changes in the matching.

2.2.2  |  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 (Figure 4).

In the top right view (Figure 4C), again an overlay of the z-slices is visualised together with the filaments. The filaments are coloured according to the matches. Automatic matches are depicted by colours ranging from bright orange to red and manual ones in shades of green. The colours are taken from ColorBrewer.org. The software also supports a colour scheme (magenta-green) for colour-blind people as well as the use of any diverging colourmap. The different colour shades support the distinction of different matching pairs, especially in regions with high MT density. Unmatched filaments are visualised in black. Because a filament can be matched at both ends, the colour fades out to grey from one end to the middle of the filament to avoid ambiguous colouring. In addition to the colouring, a match between two filaments is visualised by a line, which also fades out accordingly. To keep track of which filament corresponds to which tomogram, a coloured border around each filament is drawn. Again, blue is used for the filaments of the top tomogram and yellow for the ones of the bottom tomogram. The user can select two filament ends of the two tomograms 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 (Figure 4B) and a 3D view on the top left (Figure 4A). 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 centred 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 define manual matches for such cases. Similar 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 this view, the user can zoom, pan, and rotate the scene. To avoid visual clutter, the region is cut into Voronoi regions. It shows the filament ends, the user can also directly select a filament end of the bottom tomogram in the overlay view and confirm its state. To keep track of the state of open and confirmed ends, the GUI always displays their numbers.

The last view, the bottom right view (Figure 4D), is used to display the current state of the matching process. It shows the z-slice of the bottom tomogram, subdivided into Voronoi regions \(^1\) of the filament ends in the matching region. Regions that correspond to filament ends for which the user has confirmed the state are displayed in a bright colour. 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 a dark colour. The map always displays the same view for the bottom tomogram as in the overlay view. By selecting a region in this map, the stepper jumps automatically to the corresponding microtubule end. Hence, the map can help to evaluate the automatic matching and to steer the manual investigations. Especially for large data, regions in the tomogram that need further attention can thus be identified quickly.

Within the fourth view, again the filament endpoint 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 of the iterative improvements. A new histogram is added after each recomputation of the automatic matching; the previous ones are kept for comparison. With this, the user can quickly evaluate the impact of the previous manual editing.

### 2.2.3 Construction of stacks

When the filament ends of all section interfaces have been matched adequately, the matchings can be used to warp all tomograms of the whole stack and create a merged volume. For this, one tomogram is selected as base tomogram. Hence, this tomogram will not be warped. The tomograms above and below the base tomogram are warped based on the landmarks of the corresponding interfaces. The warped tomograms are then resampled into a single volume containing all tomograms of the whole stack. 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 tomogram ID of the point. If the quality of the

---

\(^1\) The Voronoi region of a filament end covers the whole area that is closer to this filament end than to any other filament end.
tomograms and the distribution of the filaments is more or less uniform over the full stack, the middle tomogram should be ideal as a base section to avoid accumulating the warping error over more interfaces than necessary. Otherwise, one should select a tomogram close to the middle of the stack with the best quality and filament distribution.

The GUI has a dedicated parameter section for the creation of the full stack (Figure 2A, bottom section). 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-directions of the tomograms is another way to increase speed and reduce memory consumption. Both options help to quickly generate a preview of the stack or create results on systems that do not have enough RAM to merge the tomograms at full resolution.

### 2.3 Recommended workflow

For applying the **Serial Section Aligner**, we recommend the following overall workflow (Figure 2B). After creating the **Serial Section Stack** object and detecting the correct tomogram order, the section interfaces should be processed one after another. After processing a single interface, the respective sections should be sufficiently aligned and the filaments should be matched such that it is not necessary to revisit this interface. We strongly recommend to adhere to this advice, because frequent switches between section interfaces can be time consuming due to repeated loading of tomograms. When 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 both matching and alignment look good, the user should step over a fixed number of filament ends and confirm or possibly correct them. Finally, if in some regions there are very few or no filaments, the user should consider adding a few manual landmarks in those regions.

When a new section interface is addressed, an initial automatic matching is computed. It may happen, however, that this automatic matching is completely wrong. In this case, the user should completely clear the matching. Now, there are two possibilities to continue, which can also be combined. First, the user sets a few manual landmarks in alignment mode and reruns the automatic matching. Second, the user manually matches a few filaments, which also results in new landmarks, and reruns 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 neighbourhood 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 Figure 5A, C, where the automatic matching failed locally, and the placement of two manual matches helped to improve the automatic matching for the complete region (Figure 5B, D). Finally, the same procedure is followed as described above for a direct successful automatic matching.

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

### 2.4 Technical aspects

#### 2.4.1 Stack data structure

As described above (see Section 2.1), the serial-section data is handled in a stack data structure. If \( n \) is the number of sections, then the stack stores \( n \) tomograms \( T_i \) with their 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 **Serial Section Aligner** module. During alignment and matching, only two tomograms are loaded into RAM at the same 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, landmarks always occur as pairs of 2D points in x-y-space, one point for each of the neighbouring tomograms. The points mark corresponding locations on the two sides of an interface and define an x-y-transformation that warps one tomogram into the space of the neighbouring one such that the tomograms and filaments are aligned. Because the tomograms 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 coloured 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 $b \in F_{i+1}$, where 0 denotes the start of a filament and 1 denotes the end. Similar to the landmarks, each match has a type attribute that is either manual or automatic.

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

2.4.2 Slice rendering

The rendering of tomogram slices is essential for the visual assessment of the filament alignment and matching. Our tool supports fully interactive visualisation of top- and side-view slices for both original and warped tomograms.

Let us first consider the case without warping. The cost for rendering a slice that is computed in tomogram space scales with the tomogram size. If the dimensions of the tomograms are large, it can be difficult 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 far away from the camera so that the size of a voxel is much smaller than a pixel. Hence, small details will not be visible, and it would be inefficient to render a slice at full resolution in tomogram space. Instead, we render slices in view space. For this, 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 explicitly switch between different slice qualities. For high quality, the full view resolution is considered, for medium quality, half the resolution, and for low quality, only one fourth of the view resolution. This can further increase the interactivity, especially on large screens. The effect of changing the rendering quality is shown in the Video from 1:29-1:46. Note that the ‘Advanced’ option of the Serial Section Aligner module needs to be switched on to see these rendering options.

Next, let us consider 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 using the moving least squares approach by Schaefer et al.\textsuperscript{15} For a large number of landmarks, real-time rendering of the slices might not be possible, 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 apply a 2D grid in screen space. Then, for each grid point, the corresponding 3D point on the slice is detected and only the grid points are warped. For all points within a grid cell, bilinear interpolation of the warped cell is applied to compute the warped position (Figure 6). This accelerates the rendering of the warped slice, so 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.

We can further increase the rendering performance for the warped slices using an approximation of the moving least squares approach. In its original implementation, 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.

2.4.3 Landmark rendering

Each landmark is visualised in the alignment mode as a circle on the z-slice (Figure 3A, B). 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 user interaction. For each circle, a single vertex is generated that is then extended to a quad in the geometry shader on the GPU. The centre of the quad is the centre of the landmark, and the sides equal the diameter 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 coloured according to the landmark type. In addition, the transparency of the colour is increased for fragments close to the centre of the circle. This allows the user to see the image feature that is related to the landmark.

2.4.4 Filament rendering

As mentioned in Section 2.1, each filament is represented as a piecewise linear curve, given by a set of points $p_1, \ldots, p_n$. Each filament in our reconstructions is represented by a sphere at each point $p_i$ and a cylinder between two points $p_i$ and $p_{i+1}$ with constant radius of both the spheres and the cylinders. To display the filaments within the selected slices in the tomogram, only the intersection of the spheres and cylinders with the slice is visualised. 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. 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. Ray casting is performed in the fragment shader. Similarly, for each cylinder, two 3D vertices are created for the start and endpoints, respectively. Then in the geometry shader, a quad is created that encloses the cylinder after projection. Finally, ray casting is used in the fragment shader to find the intersection of the view ray with the cylinder.

To quickly update the filament visualisation 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 visualised in a second step.

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 neighbouring 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 drastically increases the visual quality and helps to distinguish between slice data and filaments or landmarks.

2.4.5 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 (Figure 7). 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 determined from the pre-computed Voronoi diagram. Then the state of the end is used to modulate the brightness of the intensity value, as described previously in Section 2.2.2. An illustration of the filament end state map is displayed in Figure 7.

3 RESULTS

3.1 Matching and alignment examples

Using the described tools, more than 20 data sets were assembled so far (Table 1) by more than 10 different users. The data sets were obtained from different specimens, different types of spindles (i.e., mitotic vs. meiotic ones), and different stages of cell division, as well as from both wild-type and mutant samples. In Figure 8, three of these spindles are shown as reconstructed tomogram stacks with stitched MTs. These examples demonstrate the structural diversity of microtubule-based spindles.

In the following paragraphs, three challenging datasets are described in detail. For the first two, experimental problems had to be tackled, such as damaged sections or poor image quality. The third example represents the largest spindle that has been reconstructed so far. We used it as benchmark for our system.

3.1.1 Mitotic spindle at prometaphase

In this phase of mitotic cell division in C. elegans, numerous MTs emenate from the centrosomes and enter the pronuclei through the fragmented pronuclear membranes to make contact to the chromosomes. The sample stack contains a region around the pronuclei and consists of seven sections with 300-500 MTs per section (Table 1; T0354-1_1). Due to the field of view with the nucleus in the centre, most of the MTs are close to the image
### TABLE 1  Summary of data sets as used throughout this study

| Sample                              | Data set (Ref) (in this paper) | Montage | No. of sections | Volume of reconstruction ($\mu$m$^3$) | Size of data set (GB) | No. of MTs |
|-------------------------------------|--------------------------------|---------|-----------------|--------------------------------------|-----------------------|------------|
| C. elegans, female meiosis          | T0208-1$^{17,19,20}$          | 2 × 1   | 20              | 91.0                                 | 11.4                  | 3723       |
| C. elegans, female meiosis          | T0243_4$^{17,19,20}$ (Figure 8A) | 1 × 1   | 18              | 64.1                                 | 6.2                   | 6981       |
| C. elegans, first mitosis           | T0252$^2$                     | 2 × 3   | 24              | 107.7                                | 13.6                  | 3581       |
| C. elegans, first mitosis           | T0265-21 (Figure 8C)          | 2 × 2 × 3 | 23              | 977.5                                | 79.3                  | 27051      |
| C. elegans, first mitosis           | T0265-41$^3$                  | 2 × 3   | 15              | 229.4                                | 28.7                  | 11801      |
| C. elegans, first mitosis           | T0265-42$^{17,19}$            | 2 × 3   | 25              | 832.1                                | 93.1                  | 12205      |
| C. elegans, first mitosis           | T0265$^5$                     | 2 × 3   | 15              | 516.2                                | 36.4                  | 8774       |
| C. elegans, female meiosis          | T0275-10$^{17,19}$            | 1 × 1   | 14              | 108.65                               | 8.3                   | 1511       |
| C. elegans, first mitosis           | T0354-1_1$^5$                 | 2 × 2   | 7               | 171.9                                | 14.4                  | 1868       |
| C. elegans, male meiosis$^a$        | T0391-1$^{18}$                | 1 × 1   | 14              | 63.8                                 | 7.6                   | 1729       |
| C. elegans, male meiosis$^a$        | T0391-2$^{18}$                | 1 × 2   | 11              | 94.4                                 | 9.1                   | 2406       |
| C. elegans, male meiosis$^a$        | T0391-3$^{18}$                | 1 × 1   | 14              | 57.2                                 | 7.6                   | 1689       |
| C. elegans, male meiosis$^a$        | T0391-4$^{18}$                | 1 × 1   | 25              | 116.5                                | 15.6                  | 2051       |
| C. elegans, male meiosis$^a$        | T0391-5$^{18}$                | 1 × 2   | 14              | 120.1                                | 11.8                  | 1405       |
| C. elegans, male meiosis$^a$        | T0391-6$^{18}$                | 1 × 2   | 17              | 181.7                                | 16.1                  | 1540       |
| C. elegans, male meiosis$^a$        | T0391-7$^{18}$ (Figure 8B)    | 1 × 2   | 11              | 113.0                                | 9.8                   | 1403       |
| C. elegans, male meiosis$^a$        | T0391-8$^{18}$                | 1 × 2   | 30              | 248.3                                | 32.7                  | 1881       |
| C. elegans, male meiosis$^a$        | T0391-9$^{18}$                | 1 × 1   | 8               | 40.2                                 | 4.0                   | 893        |
| C. elegans, male meiosis$^a$        | T0391-10$^{18}$               | 1 × 1   | 4               | 18.8                                 | 1.4                   | 671        |
| C. elegans, male meiosis$^a$        | T0391-11$^{18}$               | 1 × 1   | 6               | 34.9                                 | 2.7                   | 246        |
| HeLa cell, mitosis                  | T0471$^9$                     | 2 × 3   | 13              | 434.7                                | 34.0                  | 5567       |

All data sets are available on request.

$^a$Data on male meiosis in C. elegans have been uploaded to the TU Dresden Open Access Repository and Archive system (OpARA) and are available as open access: https://doi.org/10.25532/OPARA-56.
boundaries in x-y. Hence, fully automatic workflow failed (Figure 5A, C). None of the section interfaces could be aligned. The reason was most likely the uneven distribution of the MTs. 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. About 25%-35% of the matches were placed manually. Substantial manual effort was especially necessary close to the section boundaries in x and y.

3.1.2 Mitotic spindle at anaphase

This stack consists of a half spindle in early anaphase in *C. elegans* (Table 1; T0265-425). The chromatids are separated. The data set includes a half spindle with a centrosome. It consists of 25 sections with 1100-4200 MTs per section. However, one section in the middle of the stack got damaged during sample preparation, and 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 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 three sections around the damaged section is 4176, 2134 and 3774, where the one in the middle is the damaged section. It was possible to handle the two interfaces above and below the damaged section within one hour.

3.1.3 Mitotic spindle at metaphase

This stack consists of a *C. elegans* spindle in metaphase (Table 1; T0265-215). The full spindle shows the chromosomes located in the centre of the cell and attached to
a subset of MTs, called kinetochore MTs. Similar 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. In contrast to the previous two examples, the quality of the sections was 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.

3.2 | Performance

3.2.1 | 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 \((n + 1) \cdot k\) memory units are required, where \(n\) is the number of sections and \(k\) the size of the largest tomogram. For our cases, 150 GB was sufficient for the reconstruction of the largest stacks in full resolution.

3.2.2 | 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 visualisation 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 mainly on the resolution of the views. For our tests, we used a desktop computer with an Intel Xeon i7x5650 processor (6 cores) and an Nvidia GeForce GTX 1080 Ti graphics card. Amira was run on a screen with 1920 \times 1200 pixels. The view size of all four equally sized views together was 1400 \times 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. Thus, with default desktop systems, it was possible to align and match serial sections for all our use cases with interactive frame rates. Since the most computationally expensive parts are done on the CPU, like the computation of the intensity values of the slices as well as the warping of the slices and the microtubules, the CPU represents the bottleneck for the full rendering pipeline. Compared to this, the rendering of the slices and the ray casting of the MTs and landmarks is negligible.

4 | DISCUSSION

We have demonstrated that the proposed tool solves the matching and alignment of serial-section tomogram stacks for data sets of different size, and number and distribution of MTs, including the 3D reconstruction of the full stack as one single volume data set. An important design decision was that the construction of complete stacks is decomposed into smaller tasks that solve the alignment and stitching problem separately for each interface between two neighbouring 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.

4.1 | Visual design

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

Due to the low signal-to-noise ratio of 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 well with image data size. We doubt that 3D visualisations 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. Visualising the MTs as 3D lines for the full stack is inadequate for inspecting the matching at section interfaces. Due to packing density, navigation in sets of 3D filamentous structures can be difficult. However, if the 3D visualisation 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 is difficult in a single slice. On the other hand, showing MT ends as overlays of two \(z\)-slices allows the user to easily detect patterns in both sections, which can be more difficult in a 3D visualisation 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 the rendering of 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 one to perceive the slope of the MTs, which helps to detect corresponding ends.

The map that depicts the MT 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 and 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 visualised a bit darker, the others show additional information such as enemies in the regions. In our case, the bottom section is initialised as an undiscovered map. Automatic unconfirmed MT ends correspond to 'visited' regions. The confirmed MT ends correspond to regions 'under control'.

### 4.2 Technical aspects

The design decision to restrict the workflow to processing interfaces between two neighbouring 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 visualisation could be quite complex and often requires advanced out-of-core rendering techniques. Since we restricted the design to a single interface, simple and efficient slice visualisations 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 (Result – Technical aspects – Section 2.4.2). We expect that due to this design even larger tomogram sizes can be handled without major modifications. Because most visualisations are based on screen-space, the aligner tool should scale reasonably well with data size. The integration of alternative automatic alignment and matching algorithms is straightforward, since the design does not use any advanced data structures.

The top left view in alignment mode (Figure 3A) is rotated and translated to approximately match the warping applied to the overlay viewer. We tested 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 less 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.

### 4.3 Practical aspects

The interactive visual feedback that shows the current state of alignment and matching for the section interfaces was particularly important to assess the quality of the results. For the alignment, we considered our initial design as sufficient. For the matching, however, we concluded that a stepping tool to confirm MT ends and the visualization of the MT end state map would be needed.

For large number of MTs, it is 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, we 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.

Since it is unfeasible to check all MT ends for large numbers of MTs, it is expected that some errors will remain in the final matching and hence the final alignment. Therefore, a trade-off has to be made between efficiency and accuracy. Checking a certain number of MT ends (about 10-25) after being visually satisfied with the alignment should give the user an idea about the expected quality of the result. According to previously published data, approximately 5% of MTs needed to be corrected. These corrections included the manual tracing of undetected MTs, the connection of MTs and deletions of tracing artefacts (eg membranes of vesicles).

Application of our approach indicated that the side view is 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 intersection of the two lines. Thus, it is easier to inspect.

Originally, the automatically computed landmarks were maintained separately from the matching process. The coherent point drift algorithm could be applied completely independently from the matching, as in the 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 did not produce a reasonable result. They rarely modified the automatic landmarks. For this reason, we decided to compute the initial automatic matching in a black box. Note that the initial automatic matching is only computed if neither automatic nor manual landmarks are available (during initialisation or when a user decides to delete all landmarks). We observed that if the alignment does not produce a good result, neither does the matching, 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 need for an 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 action. Thus, the manual and automated matching can be applied iteratively to progress towards the best achievable stitching result. It was possible to create reasonable solutions even for cases where the fully automatic stitching process failed.

5 CONCLUSIONS

In this paper, we have presented a tool to align serial tomograms in combination with the stitching of filamentous structures, specifically MTs in reconstructions of both mitotic and meiotic spindles. This tool allows users to visually validate the stitching process and support the automatic algorithms by placing manual landmarks and matches. Cases, in which the fully automatic workflow fails, can be solved with a reasonable investment of manual labour because the user can steer the whole process of stitching a stack of serial tomograms.

While stitching sections, minor errors of the tracing are often identified. For example, falsely traced MTs or erroneously connected MTs could be detected. Currently, it is not possible to edit the MT geometry with our tool. Users need to improve the MT tracing either before or after the stitching. It would be helpful to add such a functionality. This would allow users to immediately correct errors in the process of MT segmentation. For example, operations would be possible to add, split, remove, or merge MTs.

ACKNOWLEDGEMENTS

We are grateful to T. Fürstenhaupt (EM facility, Max Planck Institute of Molecular Cell Biology and Genetics, MPI-CBG) for continuous technical support. We also would like to thank I. Lantzsch, E. Szentgyörgyi and M. Kirchner for providing spindle data sets and comprehensive user feedback.

Open access funding enabled and organized by Projekt DEAL.

AVAILABILITY AND FUTURE DIRECTIONS

The described tool is available at https://www.zib.de/software/serial-section-aligner as an Amira custom module. The source code is written in C++ and OpenGL and available at https://github.com/zibamira/SerialSectionAligner. It can be used free of charge in accordance with the license given there. To execute the tool, a commercial version of Amira (version 2020.3) is required. A free demo version can usually be obtained from Thermo Fisher Scientific upon request for a limited duration. As stated previously, performance tests were carried out on a desktop computer with an Intel Xeon i7×5650 processor (6 cores) and an Nvidia GeForce GTX 1080 Ti graphics card. This hardware was sufficient to perform all interactive data processing at real time. 128 GB RAM were required to process the final stitching of the larger serial-section tomogram stacks given in Table 1. If larger data sets are to be processed, it is recommended to increase the RAM. Moreover, a larger number of CPU cores could further improve processing performance.

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-direction 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, thus improving the overall alignment and matching procedure. It should be possible to compute good cropping parameters automatically from confirmed matches. In the future, this will need to be implemented in a version of the software package to come.

AUTHOR CONTRIBUTIONS

Conceptualisation: NL, VJD, SP, TMR, DB.
Data curation: NL, GF, RK, SR.
Formal analysis: NL, FNB, VJD.
Funding acquisition: SP, TMR.
Investigation: NL, GF, RK, SR.
Methodology: NL, VJD, SP, DB.
Project administration: SP, TMR.
Resources: SP, TMR.
Software: NL, FNB, SP, DB.
Supervision: SP, DB.
Validation: GF, RK, SR.
Visualisation: NL, SP, DB.
Writing – original draft: NL, SP.
Writing – review & editing: NL, VJD, FNB, SP, GF, RK, SR, TMR, DB.

ORCID
Norbert Lindow https://orcid.org/0000-0001-5143-2573
Florian N. Brünig https://orcid.org/0000-0001-8583-6488
Vincent J. Dercksen https://orcid.org/0000-0002-9878-0548
Gunar Fabig https://orcid.org/0000-0003-3017-0978
Stefanie Redemann https://orcid.org/0000-0003-2334-7309
Thomas Müller-Reichert https://orcid.org/0000-0003-0203-1436
Daniel Baum https://orcid.org/0000-0003-1550-7245

REFERENCES
1. Muller-Reichert, T., Kiewisz, R., & Redemann, S. (2018). Mitotic spindles revisited—New insights from 3D electron microscopy. Journal of Cell Science, 131(3), https://doi.org/10.1242/jcs.211383.
2. Redemann, S., Furthauer, S., Shelley, M., & Muller-Reichert, T. (2019). Current approaches for the analysis of spindle organization. Current Opinion in Structural Biology, 58, 269–277. Epub 2019/07/08. https://doi.org/10.1016/j.sbi.2019.05.023. PubMed PMID: 31279499.
3. Weber, B., Greenan, G., Prohaska, S., Baum, D., Hege, H.-C., Müller-Reichert, T., … Verbavatz, J. M. (2012). Automated tracing of microtubules in electron tomograms of plastic embedded samples of C. elegans embryos. Journal of Structural Biology, 178, 129–138.
4. Weber, B., Tranfield, E. M., Hoog, J. L., Baum, D., Antony, C., Hyman, T., … Prohaska, S. (2014). Automated stitching of microtubule centerlines across serial electron tomograms. Plos One, 9(12), e113222. https://doi.org/10.1371/journal.pone.0113222. PubMed PMID: 25438148; PubMed Central PMCID: PMC4249889.
5. Redemann, S., Baumgart, J., Lindow, N., Shelley, M., Nazockdast, E., Kratz, A., … Müller-Reichert, T. (2017). C. elegans chromosomes connect to centrosomes by anchoring into the spindle network. Nature Communications, 8, 15288. https://doi.org/10.1038/ncomms15288. PubMed PMID: 28492281.
6. Kremer, J. R., Mastronarde, D. N., & McIntosh, J. R. (1996). Computer visualization of three-dimensional image data using IMOD. Journal of Structural Biology, 116(1), 71–76. PubMed PMID: 8742726.
7. Anderson, J. R., Jones, B. W., Yang, J. H., Shaw, M. V., Watt, C. B., Koshevoy, P., … Marc, R. E. (2009). A computational framework for ultrastructural mapping of neural circuitry. Plos Biology, 7(3), e1000074. Epub 2009/10/27. https://doi.org/10.1371/journal.pbio.1000074. PubMed PMID: 1985814; PubMed Central PMCID: PMC2661966.
8. Berlanga, M. L., Phan, S., Bushong, E. A., Wu, S., Kwon, O., Phung, B. S., … Ellisman, M. H. (2011). Three-dimensional reconstruction of serial mouse brain sections: Solution for flattening high-resolution-large-scale mosaics. Front Neuronaat, 5, 17. Epub 2011/06/02. https://doi.org/10.3389/fnana.2011.00017. PubMed PMID: 21629828; PubMed Central PMCID: PMCPMC3096995.
9. Cardona, A., Saalfeld, S., Schindelin, J., Arganda-Carreras, I., Preibisch, S., Longair, M., … Douglas, R. J. (2012). TrakEM2 software for neural circuit reconstruction. Plos One, 7(6), e38011. Epub 2012/06/23. https://doi.org/10.1371/journal.pone.0038011. PubMed PMID: 22723842; PubMed Central PMCID: PMCPMC3378562.
10. Beyer, J., Hadwiger, M., Al-Awami, A., Jeong, W. K., Kasthuri, N., Lichtman, J. W., & Pfister, H. (2013). Exploring the connectome: Petascale visualization of microscopy data streams. IEEE Computer Graphics and Applications, 33(4), 50–61. Epub 2014/05/09. https://doi.org/10.1109/MCG.2013.55. PubMed PMID: 24808059; PubMed Central PMCID: PMCPMC4296712.
11. Al-Awami, A. K., Beyer, J., Haehn, D., Kasthuri, N., Lichtman, J. W., Pfister, H., & Hadwiger, M. (2016). NeuroBlocks – Visual tracking of segmentation and proofreading for large connectomics projects. IEEE Transactions on Visualization and Computer Graphics, 22(1), 738–746. Epub 2015/11/04. https://doi.org/10.1109/TVCG.2015.2467441. PubMed PMID: 26529725.
12. Dercksen, V. J., Hege, H. C., & Oberlaender, M. (2014). The Filament Editor: An interactive software environment for visualization, proof-editing and analysis of 3D neuron morphology. Neuroinformatics, 12(2), 325–339. Epub 2013/12/11. https://doi.org/10.1007/s12021-013-9213-2. PubMed PMID: 24323305.
13. Stalling, D., Westerhoff, M., & Hege, H.-C. (2005). Amira: A highly interactive system for visual data analysis. In: C. D. Hansen & C. R. Johnson (Eds.). The visualization handbook (pp. 749–767). Burlington, MA, USA: Elsevier.
14. Harrower, M., & Brewer, C. A. (2003). ColorBrewer.org: An online tool for selecting colour schemes for maps. Cartographic Journal, 40, 27–37.
15. Schaefer, S., McPhail, T., & Warren, J. D. (2006). Image deformation using moving least squares. ACM Trans Graph, 25(3), 533–540.
16. Sigg C., Weyrich T., Botsch M., & Gross M. (eds.). (2006). GPU-based ray-casting of quadratic surfaces. Eurographics Symposium on Point-Based Graphics, Boston, The Eurographics Association.
17. Redemann, S., Lantzsch, I., Lindow, N., Prohaska, S., Strako, M., & Muller-Reichert, T. (2018). A switch in microtubule orientation during C. elegans meiosis. Current Biology, 28(18), 2991–2997. https://doi.org/10.1016/j.cub.2018.07.012. PubMed PMID: 30197085.
18. Fabig, G., Kiewisz, R., Lindow, N., Powers, J. A., Cota, V., Quintanilla, L. J., … Müller-Reichert, T. (2020). Male meiotic spindle
features that efficiently segregate paired and lagging chromosomes. *Elife*, 9. Epub 2020/03/10. [https://doi.org/10.7554/eLife.50988](https://doi.org/10.7554/eLife.50988). PubMed PMID: 32149606.

19. Yu, C.-H., Redemann, S., Wu, H.-Y., Kiewisz, R., Yoo, T. Y., Conway, W., … Needleman, D. (2019). Central-spindle microtubules are strongly coupled to chromosomes during both anaphase A and anaphase B. *Molecular Biology of the Cell*, 30(19), 2503–2514. [https://doi.org/10.1091/mbc.E19-01-0074](https://doi.org/10.1091/mbc.E19-01-0074). Epub 2019 Jul 24. PMID: 31339442.

20. Lantzsch I., Yu C.-H., Chen Y.-Z., Zimyanin V., Yazdkhasti H., Lindow N., Szentgyörgyi Erik, Pani A. M, Prohaska S., Srayko M., Fürthauer Sebastian, Redemann S. (2021) Microtubule re-organization during female meiosis in *C. elegans*. *Elife*, 10. [https://doi.org/10.7554/elife.58903](https://doi.org/10.7554/elife.58903).

**SUPPORTING INFORMATION**

Additional supporting information may be found online in the Supporting Information section at the end of the article.

**How to cite this article**: Lindow, N., Brünig, F. N., Dercksen, V. J., Fabig, G., Kiewisz, R., Redemann, S., Müller-Reichert, T., Prohaska, S., & Baum, D. (2021). Semi-automatic stitching of filamentous structures in image stacks from serial-section electron tomography. *Journal of Microscopy*, 1-20. [https://doi.org/10.1111/jmi.13039](https://doi.org/10.1111/jmi.13039)