Martin Luther University Halle-Wittenberg
Institute of Computer Science
Pattern Recognition and Bioinformatics

User Guide

rhizoTrak

Version 0.4 – January 7, 2019

written by

The *rhizoTrak* Development Team

Birgit Möller Stefan Posch

Tino Schmidt Axel Zieschank

Licensing information:

This manual is part of ***rhizoTrak***, a software tool for the manual annotation of (mini)rhizotron images.

rhizoTrak builds upon the *trakEM2* project for neural circuit reconstruction initiated by Albert Cardona in 2005. Large parts of the *rhizoTrak* code have directly been copied or are derived from the code basis of *trakEM2*.

trakEM2 is licensed under GNU General Public License version 3. For more information on *trakEM2* visit its homepage at https://imagej.net/TrakEM2.

More information about *rhizoTrak* can be found at
https://github.com/prbio-hub/rhizoTrak/wiki/.

***rhizoTrak* is a project at the Martin Luther University Halle-Wittenberg.**

Institution:

Institute of Computer Science
Faculty of Natural Sciences III
Martin Luther University Halle-Wittenberg
Von-Seckendorff-Platz 1, 06120 Halle, Germany

Contact: rhizoTrak@informatik.uni-halle.de
Webpage: https://github.com/prbio-hub/rhizoTrak/wiki/

[1] http://www.gnu.org/licenses/gpl-3.0.html
[2] http://www.fsf.org/

Contents

1 Overview

Roots provide vital functions for terrestrial plants, such as anchorage, resource uptake, transport, and storage, which are intrinsically linked to root growth and mortality. Despite its importance, root turnover is much less studied than temporal dynamics of aboveground organs, especially in situ. Minirhizotron images enable non-destructive measurements of standing root length, diameter, production, and mortality with high temporal resolution. However, extracting data from minirhizotron images requires extensive effort for manual root annotation.

rhizoTrak is a free tool for flexible and efficient manual annotation of complex time-series minirhizotron images. It supports import and processing of the complete set of images of a time series at once. This allows for easy browsing through the data and enables to annotate roots by simultaneously considering images of different time points. Temporal links between roots in different images are explicitly represented in *rhizoTrak* by grouping annotation objects via so-called connectors. Annotation data of one time point can easily be transferred as initialization to the subsequent image whereas connectors between corresponding root annotations are automatically established. Contrary to many other tools *rhizoTrak* inherently supports representation of branching roots by modelling roots as treelines. Each node is augmented with an individual radius, and segments can be annotated with user-definable status labels. For root treelines a large collection of editing operations is available including split and merge, and many of these operations are easily accessible via keyboard shortcuts. Finally, annotations can be imported and exported in RSML format [3], and annotations may be written to csv files. *rhizoTrak* builds significantly on TrakEM2 [1] and is realized as a Fiji plugin [4] available for Windows, Linux, and MacOS.

2 Installation

rhizoTrak is implemented as a plugin for the open source image processing program **Fiji**. You can download Fiji from `https://fiji.sc/` and follow the installation instructions of Fiji. Run Fiji by opening the application file provided in the Fiji directory (on Windows/MacOS) or by executing the available shell scripts (Linux).

For installing **rhizoTrak** within Fiji do the following:

1. Make sure that your IP address is registered in Halle
   (if that's not the case contact birgit.moeller@informatik.uni-halle.de).
2. Run Fiji.
3. Navigate to the update function via "Help" → "Update..." from the Fiji menu.
4. Click the button "Manage update sites" in the ImageJ Updater.

5. Add the local MiToBo update site

   - Edit the URL: http://www2.informatik.uni-halle.de/agprbio/mtbuser/
   - Double click onto the name (*New*) and change it to *MiToBo local* or similar

   The fields "Host" and "Directory on Host" may be left empty.
   Check the new entry.

6. Make sure that the entry for MiToBo (which refers to the official site) is **not** checked, i.e. activated.
   Activating the local and the official update sites simultaneously in the same Fiji installation will most likely result in crashes of Fiji!

7. Click the button "Close".
   Fiji will now display a message that there are some new files to install.
   Click the button "Apply changes" to install the files, then restart Fiji.

Now you are ready to use *rhizoTrak*. You should run the Fiji updater from time to time to get updates of *rhizoTrak*.

3 Quick start

In this section we will guide you through setting up your first *rhizoTrak* project and introduce its basic workflow. We will start with creating the project, importing an image, annotating a root in the image by adding a treeline, and finally extracting experimental data. For more details on the individual steps please refer to the Sections 4 and 5.

Create a *rhizoTrak* project *rhizoTrak* organizes the annotation of time series images scanned at the same location or the annotation of one individual image in a *project*. There are several ways to create a project. The easiest way is to use the default structure which is sufficient for most cases. For more detail on project creation and structures see Subsections 4.1 and 5.6.

Now let's start!

1. Open Fiji and start the *rhizoTrak* plugin via
   `Plugins -> rhizoTrak-> New Project (default rootstack)`

2. In the "Name and location of new project" dialogue, choose a directory and then enter a filename where you want to store this new *rhizoTrak* project. If the filename does not end with `.xml`, this extension is added to the selected filename. See Subsection 4.1 for details on a suitable choice of the project directory.

3. Next *rhizoTrak* asks for a "Status file". We will use the default status labels and just click "Cancel" to complete the process. For more information on status label see Subsection 5.3.

This will create two windows:

1. the main project window entitled "rhizoTrak" (see Figure 1) which has three parts:
   - *Template*: the project structure, see Subsection 5.6
   - *Project Objects*: the tree of all annotation objects which have been created, see Subsection 4.4
   - *Layers*: listing all layers of the project, where for each image or time point exactly one layer exists

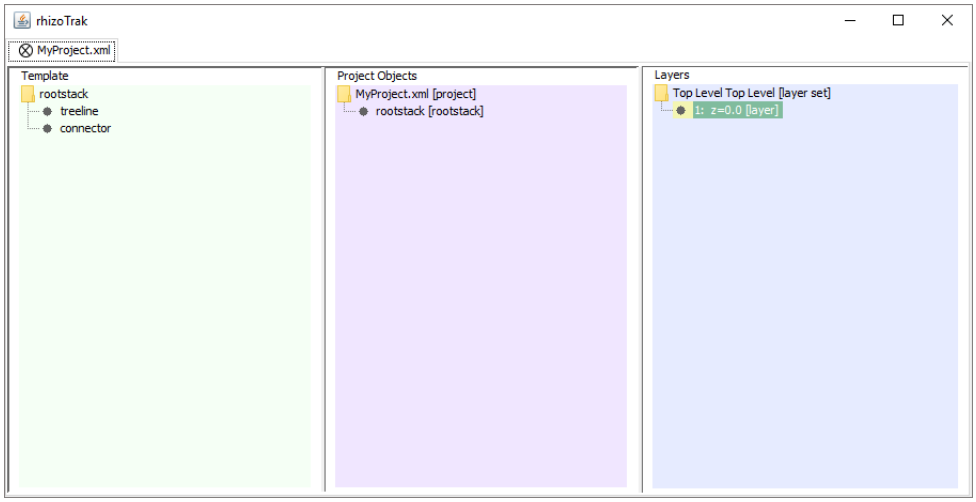


Figure 1: The main project window of *rhizoTrak*.

2. the layer window (see Figure 2) with
   - a tool bar (upper left)
   - the function tabs (below)
   - an image panel (right) which is black as no images are yet imported
   - a navigator panel (lower left) which is also black as no images are yet imported

Import the first image Now we may import our first sample image into the new project via the layer window. Go to the function tabs and select

`RhizoTrak Operations -> Load Images .`

A new window entitle "Image Loader" will open.

1. Press "Select Images", navigate to the directory where your sample image is located, and select the image.

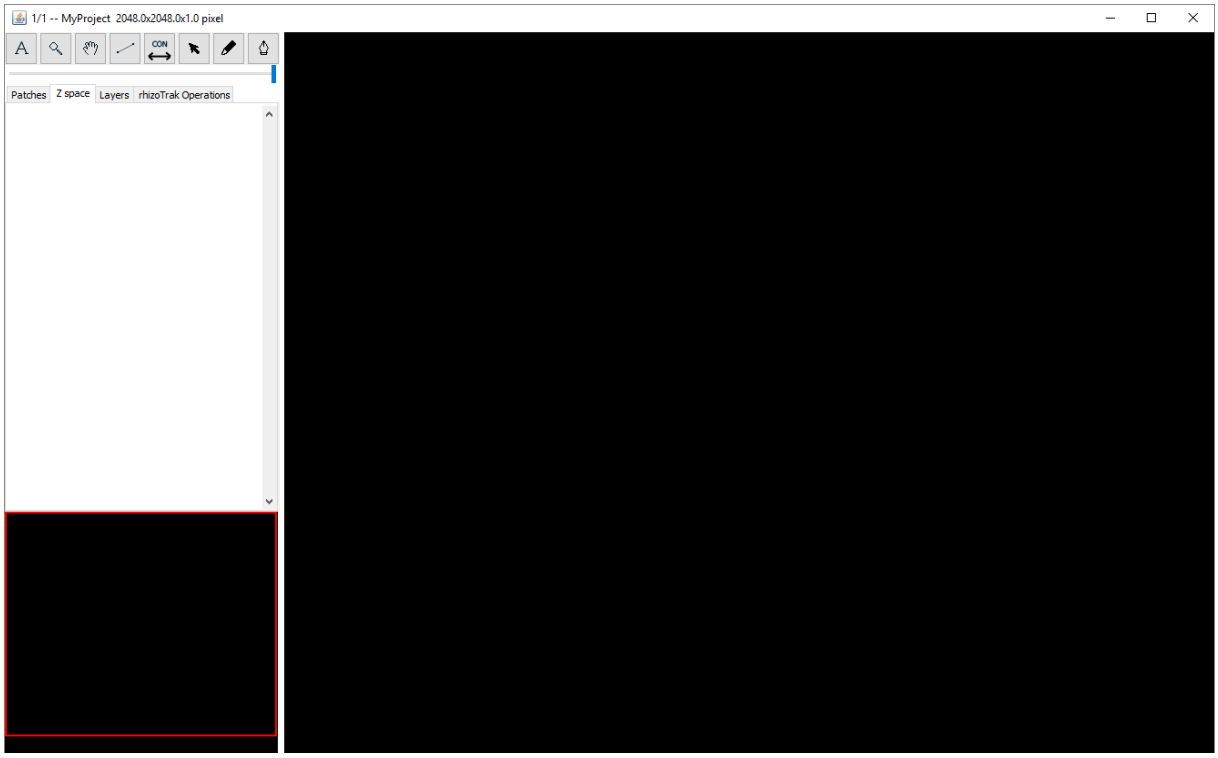


Figure 2: The layer window before importing images.

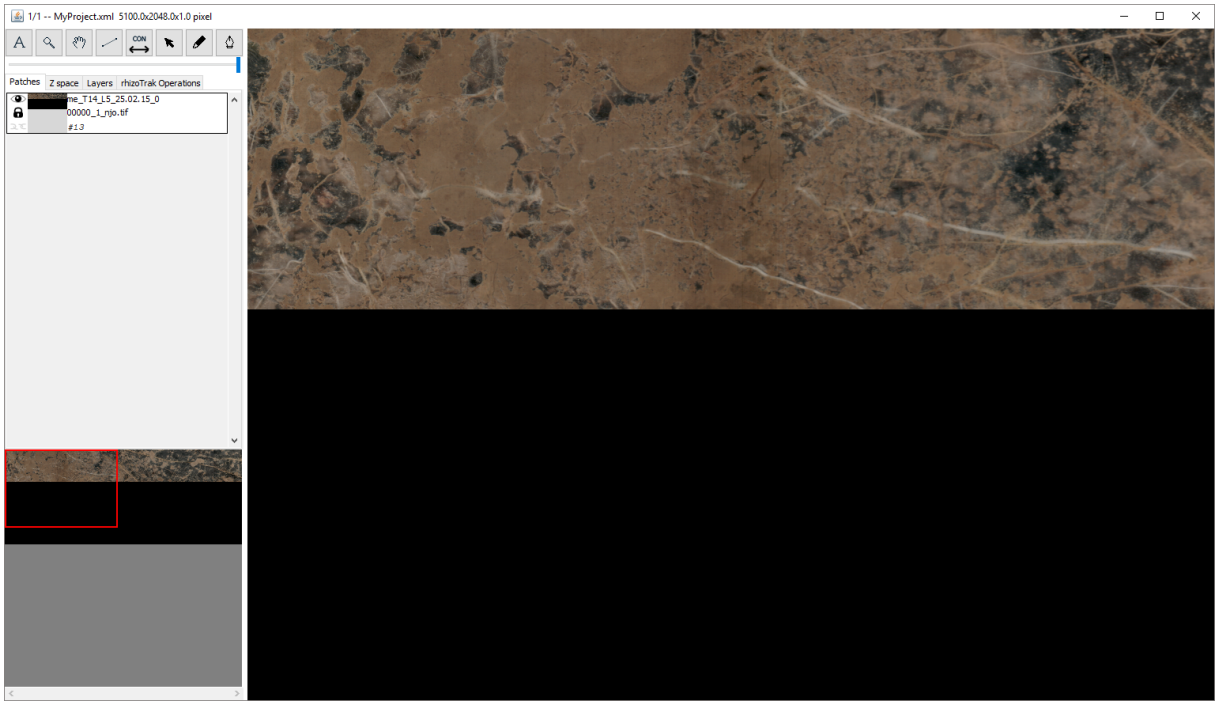


Figure 3: The layer window after importing an image.

2. Press “Import Images” to import the image.

If everything worked fine, the image panel now displays the image just imported (see Figure 3). An iconified version is displayed in the lower left corner in the navigator panel. You can navigate through the image by dragging the red rectangle.

Annotating your first root In *rhizoTrak* the annotation of a root system is represented with a set of interconnected straight lines (see Figure 4 for an example). The connections need to be such that no cycles are introduced, i.e., no circular structures are built. Such a set of interconnected straight lines is called a *treeline* in *rhizoTrak*, and this term is used throughout this manual in the following. The individual straight lines of a treeline are called *segments*, while their endpoints are called *nodes*. The base of a treeline is marked with the letter `s`, the tips of a treeline with an `e`. A branching node is marked with a `Y`.

Figure 4: The layer window with an annotated root.

To add a new treeline go to the layer window, move the mouse to the image panel, and press the key `'a'`. This creates a new empty treeline. In addition, the pen-tool  from the tool bar is activated and the `Z space` is selected. The `Z space` tab lists all existing treelines and connectors. In Figure 4 is just one single treeline present[3]. While the pen tool is activated you may add new nodes to the treeline by clicking in the image area. Each click will create a node

[3]Unique Identifiers, in the example `#18` for the treeline, are assigned to all internal objects sequentially, thus the first treeline has not `#1` as might be expected.

and a new segment starting from the previously active node to the new node. The new node will become the new currently active node, and is displayed in green by default.

Further available actions (with the pen tool being active) are:

- select a new active node pressing `'g'` while hovering over the node
- moving a node via dragging
- inserting a new node on a segment via `Shift+click` on that segment
- `Shift+Ctrl+click` to delete a node
- `Shift+Ctrl+Alt+click` to delete a node and nodes downstream to the tips
- a complete treeline may be deleted in the `Z space` or in the image panel using the context menu of the treeline

trakEM2 and thus *rhizoTrak* provide a undo-function for modifications of the annotation via `Ctrl+z`. However, not all actions can be un-done in all situations. See the *trakEM2* manual also for the redo function.

Once you are finished annotating the image, save your project via right-click in the image, then selecting

```
Project -> Save
```

or alternatively by pressing `'s'` within the image panel.

To close a project just close the main project window.

Export experimental data *rhizoTrak* allows to save experimental data for your annotations to a file in CSV format. This function is provided via the functions tab

```
RhizoTrak Operations -> Write csv
```

Selecting this option will bring up a window where you can configure the output type (individual segments or aggregated experimental data within layers), the data output range (i.e. annotations for all layers or just those of the currently active layer), the unit for the measurements, and the column separator. To finally write the file just click "OK".

4 Basic Usage

In this section we will dig into the internals of *rhizoTrak*. In particular, we will explain image data and project management within *rhizoTrak*, and its functionality concerning handling images and adding annotations. Note that this manual is not an exhaustive introduction to the

complete functionality of *rhizoTrak* and *trakEM2*, respectively, but rather concentrates on specific extensions of *rhizoTrak*. Most functions and concepts inherited from *trakEM2* will not be detailed out and the reader should refer to the *trakEM2* manual[4] for further details.

4.1 *rhizoTrak* Projects

Similar to the basic concept of *trakEM2* image data and annotations are organized in projects in *rhizoTrak*. Each project is associated with one or more images and corresponding annotations. Projects allow for hierarchically structuring annotations to model semantic relations between annotation objects (see Subsection 5.6).

The initial structure of a *rhizoTrak* project is determined during creation. There are three options to create a new project:

1. `Plugins -> rhizoTrak-> New Project (default rootstack)`

   This creates the default structure which is described in Subsection 5.6.

2. `Plugins -> rhizoTrak-> New Project (from Template)`

   Here you can supply a template file to define the annotation hierarchy. This options is for advanced use (see Subsection 5.6). (If the selected template file is invalid or the selecting a template file is cancelled, an empty structure will result.)

3. `Plugins -> rhizoTrak-> New Project (blank)`

   This creates a project with an empty structure. Again, this is for advanced use.

Each project is stored in a project directory which has to be determined upon creation of the project. A common choice is the directory where all images to be annotated within the project are located, or a common super directory of all images to be imported into this project (see Figure 5). The reason for this common choice is the following: *rhizoTrak* stores the filenames and paths of the images, rather than the images itself. In case the image is located in a sub directory of the project directory, the relative locations are stored in the project file, otherwise the absolute path. In the first case it is therefore possible to move or copy the complete project directory including subdirectories to another location or computer without the necessity to manually edit the project file. Note, you should not rename or move image files after importing them into *rhizoTrak*. Otherwise it is necessary to edit the main project file (see Subsection 5.1) using a text editor.

During *rhizoTrak* project creation, the user is also asked for a "Status file". *rhizoTrak* allows for assigning individual status labels to the segments of an annotated treeline. The set of available labels can be configured by the user. An initial set of labels can be determined via the status

[4] *trakEM2* manual, https://www.ini.uzh.ch/~acardona/trakem2_manual.html

```
...
└── myRhizoTrakProjectDir
    ├── <name>.xml
    ├── <name>.rtk
    ├── Timepoint1
    │   └── exp_T24_L1_23.04.15_160848_1_me.tif
    └── Timepoint2
        └── exp_T24_L1_03.05.15_184146_2_you.tif
```

Figure 5: Example for a directory layout which all images in sub directories of the project directory

file upon creation, and be modified interactively later on (see Subsection 5.3). The file format is described in Appendix C. If no such file is necessary, this step can be skipped by clicking "Cancel". In this case the set of default labels LIVING, DEAD, DECAYED, and GAP is provided. The status file selected during creation of a project is only used as a template for the status labels and not used later on by this project. When saving the project this information is stored in the project configuration file (see Subsection 5.1). More information about status labels can be found in Subsection 5.3.

The default structure of *rhizoTrak* projects declares a top level object of type *rootstack*. A rootstack may contain an arbitrary number of objects either of type *treeline* (Section 4.4.1) or *connector* (Section 4.4.3).

4.2 The Graphical User Interface

The graphical user interface of *rhizoTrak* consists of two windows: the main project window (Figure 1) and the layer window (Figure 2). The main project window keeps track of the project structure and lists all annotation objects and layers (see Section 3). The layer window displays the images and allows to add and modify annotations.

The layer window is subdivided into four sections. The largest fraction of the window is filled by the image panel on the right which displays the image of the active layer (Section 4.2.1). In the top-left corner is the ImageJ/Fiji toolbar which offers various tools for selections and navigating in the image. Below a set of function tabs are located (Section 4.2.2).

It is possible to have more the one layer window for the same project, e.g., to compare image structures from different time points more conveniently. To open a new layer window double-click on one of the "layer"s in the right column of the main project window.

rhizoTrak features also a richer graphical user interface ("full GUI") which makes all features of *trakEM2* available to the user. In the default mode, many *trakEM2* features of the function tabs, short cuts, and items in context menus are disabled to enhance usability for typical operation when annotating roots. The mode of the graphical user interface may be configured via

`RhizoTrak Operations -> Preferences -> Configuration` .

Note, that changes will take place only upon restart of *rhizoTrak* and when the user settings are saved on closing the project.

4.2.1 Image Panel

In the image panel the image of the active layer is displayed. In the example in Figure 6 the second of two imported images is active and displayed. This is indicted with `2/2` in the beginning of the window title. For zooming in and out of the image use `Ctrl+Mousewheel` or use the `+` and `-` signs of the keyboard. Once the image is zoomed in and only a part of the image is visible in the image panel, the hand-tool  in the tool bar can be used to navigate through the image. When the hand tool is activated, simply hold the left mouse button down and drag within the image area. Alternatively use the navigator panel in the lower left corner of the layer window and move the red box to navigate within the image. A fast option to temporarily activate the hand-tool it to press the middle mouse button and drag within the image area. Releasing the middle mouse button will re-activate the previsously active tool.

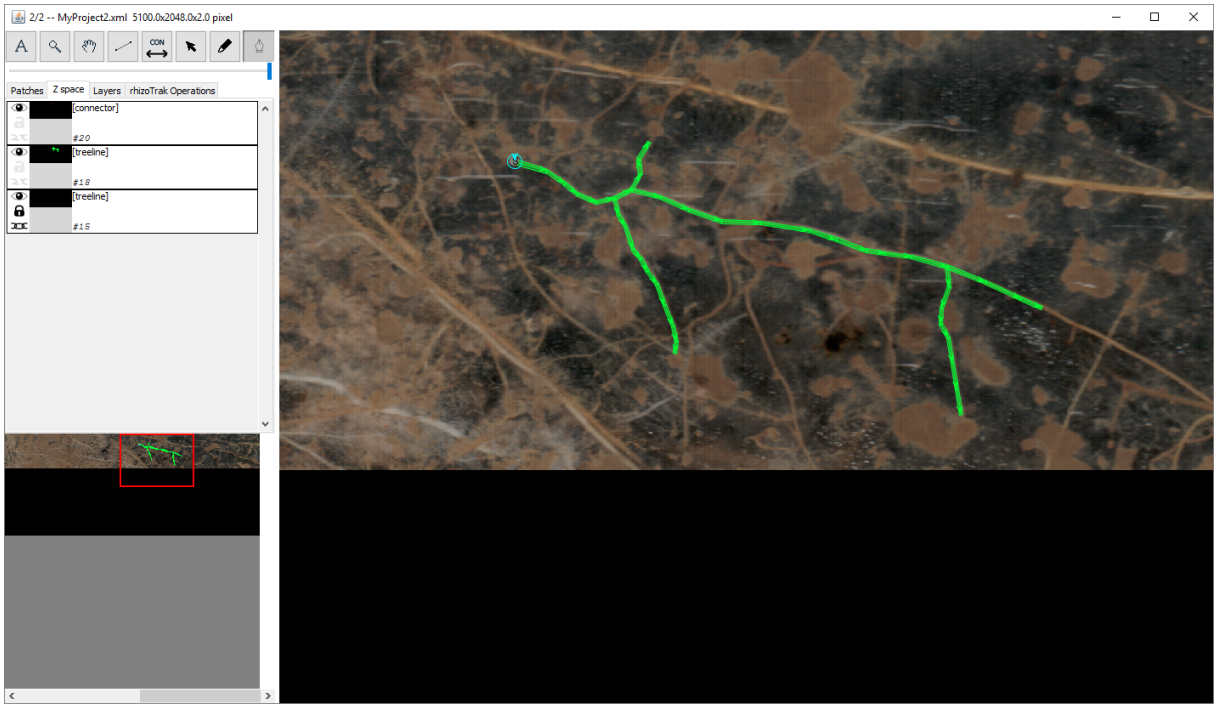


Figure 6: The layer window with two images imported.

4.2.2 Function Tabs

The set of function tabs grants access to a wide variety of functionality. In the context of *rhizoTrak* in particular the tabs denoted “Z space”, “RhizoTrak Operations”, and “Patches” are of interest:

- the `Z space` tab keeps track of all annotation objects:
  Here individual annotation objects can be selected. One can manipulate an object in a variety of ways by right clicking on the object which pops up a context menu. In addition, the tab features buttons for changing the visibility of the object as well as for locking/unlocking and linking/unlinking it. This can be accomplished via the icons on the left of the individual object items (see Figure 7). By clicking on the eye the object can be set visible or invisible, the lock in the middle allows for locking and unlocking. Likewise the bottom icon allows for linking and unlinking an object. If a treeline is locked it may not be moved at all. If it is linked, it will be moved together with its image. Note, the image may only be moved if not locked, see `Patches` tab below.

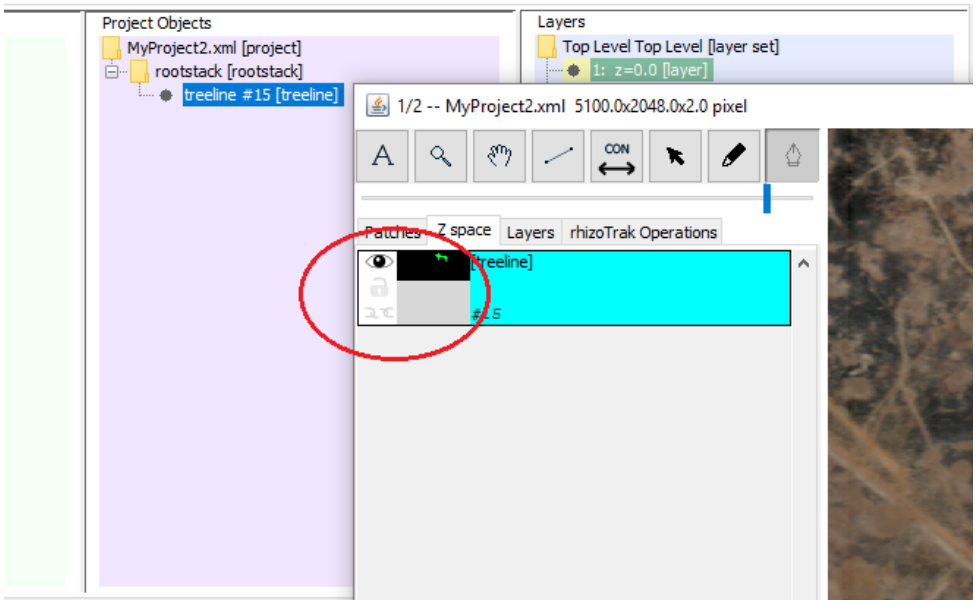


Figure 7: Tool icons on the "Z space" tab for modifying settings of individual annotation objects.

- the `RhizoTrak Operations` tab grants access to *rhizoTrak*-specific functionality:
  - `Copy`: propagate all annotations of one layer to the subsequent one, see Section 4.4.2 about handling time-series data
  - `Delete`: delete all annotated treelines from the current layer
  - `Load Images`: import of images, particularly time-series images, see Section 4.3
  - `ReadRSML`: import one or multiple RSML files, see Subsection 5.5
  - `WriteRSML`: export the current or all layers to one or several RSML files, see Subsection 5.5
  - `Write csv`: export experimental data of annotations to a csv file, see Section 4.5
  - `Conflicts`: opens editor for resolving connector conflicts and inconsistencies, see Section 5.2

  - `Preferences`: configure status labels and their display options, see Section 5.3, as well as project configurations and user settings
  - `About rhizoTrak`: shows informations on, e.g., the version of *rhizoTrak* you are using

- the `Patches` tab shows the currently selected layer along with the image name and allows to control visibility, lock or unlock, and link or unlink an image layer. If unlocked, the image may be translated with the select tool  . Translating adds an offset to the image, thus the upper left corner has not its default coordinates $(0, 0)$.
  Important note: If the image is translated, objects which are not locked and linked will not be translated together with the image, thus their relative position to the image content is lost!

The `Layers` tab may be used to overlay multiple layers in the image panel, see the Subsection "Layer overlays" in the *trakEM2* manual (https://www.ini.uzh.ch/~acardona/trakem2_manual.html#layer_overlays).

4.3 Import of Images

rhizoTrak offers a comfortable way to import a single image or images of a time series into a project via the function tab

`RhizoTrak Operations -> Load Images`

A new window will open with an initially empty list of images to import on the left, offering several options at the right (see Figure 8):

- `Select Images`: Select a set of images using a common file chooser dialog for later import.
- `Search New Images`: This is convenience function mainly for projects adhering to image naming conventions, see Appendix B. In the "Image Directory" selected (see below) all files with extension `.tif`, `.tiff`, `.jpg`, `.jpeg`, `.png`, and `.gif` are looked up and pasted into the left area (see Figure 8) for subsequent import. If images have already been imported adhering to the naming convention, only images from the same tube and new time points are considered.
- `Sort by Timepoint`: Use this button to sort the images in the list by time point automatically. This again requires the image filenames to stick to image naming conventions.
- `Import Images`: Import the selected images into the project, for each image a new layer will be added to the project in the given order adding the images after the previously imported images.
- `Select Image Directory`: Select the "Image Directory" to search for new images. On start of the project this is the project directory.

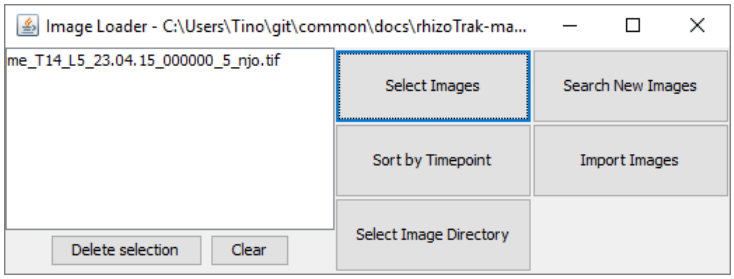


Figure 8: Window for importing single images and time series.

s

First select a set of images using `'Select Images'` and/or `'Search New Images'`. The list of these images is displayed in the import dialog on the left. Second, sort the images using the `'Sort by Timepoint'` button or by drag-and-drop in the list of images. Images from the list may be deleted using one of the buttons below the import dialog:

- `Delete selection:` Deletes the selected image file name from the list.
- `Clear:` Will clear the list.

Third, press `'Import Images'` to complete the import. Depending on the image size and number, this process may take a couple of seconds.

If everything worked fine, the layer window now displays the first of the images just imported (see Figure 3). An iconified version is displayed in the lower left corner. If multiple images from different time points have been imported, *rhizoTrak* will treat each of these as a different *layer*. You can switch between the different images, i.e. layers, by using the mouse wheel in the image panel. Switching is also possible using the keyboard shortcuts `'.'` (period) and `','` (comma). The currently displayed layer is shown in the title bar of the layer window. In Figure 6 `'2/2'` indicates, that the second of a total of two images is currently shown.

Additional images may be imported subsequently by repeating steps 1 to 3. Newly imported images will be added to the end of the layer stack.

A layer including the image and annotation objects may be removed in the *rhizoTrak* main window. In the right column entitled `Layers` right click on a layer and then choose `Delete ...` from the context menu.

4.4 Annotation of Roots

While *trakEM2* supports a variety of different types of annotation objects, two of them are of particular importance for *rhizoTrak*: *treelines* and *connectors*. Treelines are well-suited to annotate the geometry of roots, while connectors are used for linking multiple treelines in time series images referring to the same physical root.

4.4.1 Treelines

In *rhizoTrak* the annotation of a root (or root system) is represented with a set of properly interconnected polygons, called polylines in the following (see Figure 4 for an example). Such a set of interconnected polylines is called a *treeline*. The straight lines of treelines are called *segments*, while their endpoints are called *nodes*. Each node of a polyline has an associated diameter. The width of the straight segment is determined by the diameters of its nodes (see Figure 9). The base or starting point of a treeline is marked with the letter `s`, each tip of a treeline with an `e`. The diameter is depicted with a small line perpendicular to the segment if configured (see Subsection 5.4). For each segment its orientation is depicted with an arrow pointing downstream, i.e., in the direction of the tip(s). Each segment of the treeline is associated with a status and this status label is displayed abbreviated in small white boxes if the treeline is selected. The set of available status labels and corresponding abbreviations may be defined upon creation of a new project (Subsection 4.1) and changed by the user interactively using `Preferences` under `RhizoTrak Operations` (see Subsection 5.3). There are further settings to modify the way, segments are displayed. The diameter may be shown with lines, circles, or not at all. The left and right border of segments may be displayed and the segment may be filled according to its status label with the corresponding color and alpha value (see Subsection 5.4 for configuration). Pressing the key `'q'` toggles if all treelines in the current layer are to by hidden or displayed.

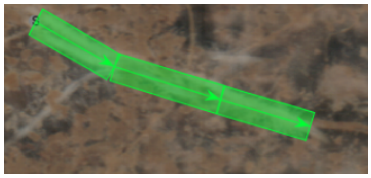


Figure 9: The width of a segment is determined by the diameters of its two nodes. The diameter is depicted with a small line perpendicular to the segment.

Adding and modifying treelines To add a new treeline to an image, go to the layer window and press `'a'` within the image panel. This will also effect activate the pen-tool  in the tool bar and the `Z space`. The new treeline will show in the list of objects in the `Z space`. While the pen tool is active, you may add nodes to the treeline by clicking in the image area. This will create a new segment starting from the previously selected node to the new node. The selected node is displayed by default in green. This color may be configured in the `Color & Visibility Panel` under `RhizoTrak Operations` (see Subsection 5.3). The diameter of the new node and the status of the new segment will be the same as the previous node and segment, respectively. To select a different node of the same treeline, click onto this node while the pen tools is still active. Alternatively you may use the key `'g'` while the mouse is pointing at the node. The later version works for any treeline (selected or not) and will also select this treeline along with the

node. Note that if nodes from different treelines are overlapping or are too close to each other and you try to select one of them, a dialogue will ask you to select a treeline. If you scroll the list of available treelines or connectors the selected object will be highlighted in the image panel. Segments of treelines may be made selectable or non-selectable depending on their status label (see Subsection 5.3).

Further actions with the pen tool being active are:

- to move a node drag it with the the left mouse button pressed
- to insert a new node to a segment `Shift+click` at the position where you want to place the node
- to adjust the diameter of a node use `Shift+Alt+Mousewheel` on the node (only for nodes of a selected treeline)
- to adjust the status of a segment use `Shift+Mousewheel` on the node where the segment ends (only for nodes of a selected treeline)
- to adjust the status of the current segment and all subsequent nodes use `Shift+Alt+Ctrl+Mousewheel` (only for nodes of a selected treeline)
- delete a node of a selected treeline with `Shift+Ctrl+click`
- to delete a node of a selected treeline and its entire subtree of nodes use `Shift+Ctrl+Alt+click`
- to add a branch of the root system select a node, e.g., using the key `'g'`, and add the first node of the branch with a left click at the intended position
- the treeline may be "re-rooted", where the selected node is is declared as the new base node using the key `'d'`

The following actions are available irrespective of the activated tool:

- a complete treeline may be deleted in the `Z space` from the context menu of the treeline
- use the key `'f'` in the image panel to toggle the display of the abbreviated status labels.
- use the key `Alt+'f'` in the image panel to toggle the display of arrow heads of the segments of treelines

Once your treeline is completed, press `'a'` to create the next treeline.

As an alternative to using the key `'g'` you may select a treeline activating the select tool  and clicking onto the treeline.

Note that for moving an entire treeline, first of all you need to unlock and unlink the treeline. This may be done via the symbols in the `Z space` tab (Figure 7) or using the context menu of the treeline. Then select the treeline and activate the select tool. The treeline can now be dragged to a new position in the image panel while holding the left mouse button pressed.

Splitting treelines A treeline can be split up into two different parts:

1. Select a node where the treeline is to be split, e.g., by pointing at the node and pressing `'g'`. The treeline will be split at the **start node of the incoming segment** of the selected node. It may seem somewhat strange to not split at the node selected. However, in case of more than one downstream segments of the selected node splitting would be ambiguous.

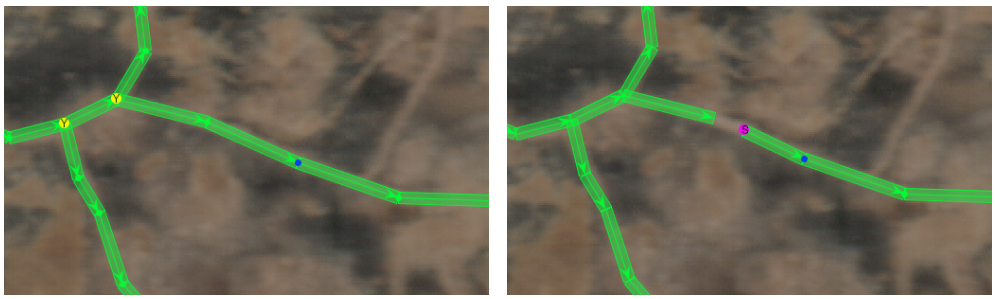
2. Right-click to open the context menu and select `Part subtree`.

3. If configured, a message window will pop-up where you need to confirm the splitting. You can configure this as part of the user specific configuration via
   `RhizoTrak Operations -> Preferences -> Configuration` .
   See also Subsection 5.1.

   Click "Yes".

4. If the treeline to be split belongs to a connector (cf. Section 4.4.3), a window will pop up and ask which of the subtrees remains as part of the connector and which should get a new connector.

The treeline will then be split in the following way. The **incoming** segment of the selected node will be detached at its upstream node and this node will be the new base node of the detached treeline. This new base node is shifted a small distance in the direction of the other node of the segment to avoid two nodes from now different treelines to have the same location. See Figure 10 for an example.



(a) (b)

Figure 10: Before (a) and after (b) the splitting of a treeline with the selected node highlighted in blue.

Merging treelines To merge two treelines in a layer use the Connector/Merge Tool 

1. select the node from the first treeline to which the second treeline is to be attached as a subtree (e.g. by pressing `'g'` while hovering over the node - see above).

2. Then activate the Connector/Merge Tool and click on the node on the second treeline which should be attached to the first node. A message window will pop-up where you need to confirm the merge operation. As for splitting actions, you may configure to be asked to confirm merging again using
   `RhizoTrak Operations -> Preferences -> Configuration` .

As a result a new segment will be added between those two selected nodes, and the second treeline will disappear from the `'Z space'` tab and become a subtree of the first treeline.

If the selected node of the second treeline is not the base node of that treeline, the topology of the second treeline is changed upon the merge. The node to be linked to the first treeline will become the base node of the second part of the complete treeline and starting with that node all outgoing segments and subtrees will become downstream segments, i.e., they might be reversed.

Finally, merging treelines can cause problems if both treelines are part of connectors (see Section 4.4.3). Sometimes such problems cannot be handled automatically and require user intervention (Section 5.2).

Tagging of nodes The nodes of treelines may be tagged with strings. The following description is taken from the *trakEM2* manual [1].
“ To tag a node:

1. Pick the “Pen” tool .
2. Put the mouse over the node.
3. Push ’t’ and then immediately any other key (like ’a’ through ’z’).

To untag, push ’shift + t + key’ instead.

The ESC key will cancel a tagging operation.

If a key doesn’t have a tag associated with it, a dialog asks to add one first. Then the node is tagged with it. If you want to define a new tag for a key that already has a tag associated with it, push ’t’ then ’0’ (zero) and then the key. A dialog pops up to associate another tag for that key.

Then, on pushing ’t’ and that key on a node, a popup menu presents the list of possible tags for that key. Notice the numeric accelerators: push the number to choose that key. ”

4.4.2 Working with Time Series

In the following we assume that images from one minirhizotron tube, acquired at different time points have been imported.

The basic workflow supported by *rhizoTrak* is to annotate the different time points in the temporal order they have been imported into *rhizoTrak*. As soon as the first time point has been annotated, all these annotations, i.e. treelines, are copied into the next time point, i.e. next layer. Copying from the currently selected layer to the next is accomplished via

`'RhizoTrak Operations' -> 'Copy'`

This also creates the information of treeline identity among time points. Internally these identities are represented as `connectors` (see Subsection 4.4.3). Subsequently the copied treelines may be modified according to the new image, including moving nodes, extending or shortening a treeline, modifying diameter and status labels for segments. Of course, treelines may be deleted completely, e.g., if a root disappeared in the new image. As an alternative, they may be annotated with an appropriate status label, e.g., DECAYED. Of course, new treelines may be added.

The function to copy all treelines will be most useful, if images of the different time points have been aligned. Aligning may be done prior to import, or using the facilities of *trakEM2* (see Registration in the *trakEM2* manual).

A single treeline may be copied to the next or an arbitrary layer via its context menu.

4.4.3 Connectors

In *rhizoTrak*, the projections of one physical root into different images of a time series are represented by one individual treeline in each of the layers. The fact, that these treelines belong to the same physical root is represented with one *connector*. The concept of connectors is inherited from *trakEM2*, however the implementation is enhanced in *rhizoTrak*.

A connector is a special treeline with one base node (as any treeline) and a set of direct child nodes of the base node. All child nodes are tip nodes, i.e., have no further child nodes. Figure 11 shows the principle way connectors are displayed. The tip nodes are marked with an 'x' if the connector is selected. The status labels of all segment of a connector have the internal status label CONNECTOR which for a selected connector is displayed with the abbreviation 'c'. For each tip node an additional circle around the node is displayed to enhance visibility (see below). The color of the circle and the segments of connectors can be configured by the user in the Color&Visibility panel, see Subsection 5.3. As the treelines associated to the same connector are in different layers in a consistent situation (see Subsection 5.2 for details), a configuration as depicted in Figure 11 will typically not arise.

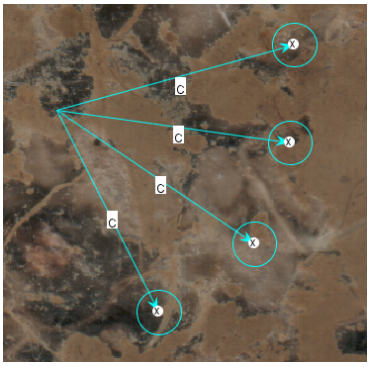


Figure 11: An example of the general form of a connector with no treeline attached to it. For a selected connector the tip nodes are marked with 'x' and the status label CONNECTOR is displayed with the abbreviation 'c'.

For each treeline which is a member of a connector, a node within this connector exists. The position of this connector node is the position of the base node of the treeline. In most cases a connector contains treelines from different layers (see also Subsection 5.2) and typically the base nodes of these treelines have the same geometric position. Thus the segments of connectors are usually not visible besides the arrow head (see Figure 12). In addition a circle is drawn around the connector node in the connector segment color. Both the arrow head and the circle are drawn with the color of CONNECTOR segments.

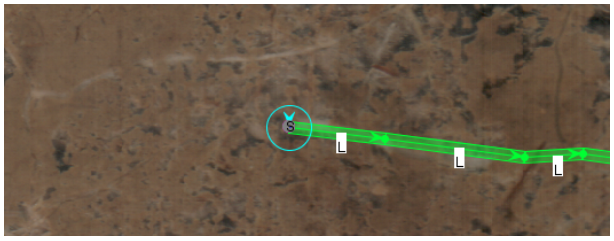


Figure 12: Typical visualization of the connector a treeline is member of. The connector is depicted with a circle around the base node of the treeline and an arrow head pointing at this base node.

If a connector is selected, all treelines belonging to this connector are highlighted with the "highlight color 1" (magenta in the examples shown). Thus scrolling through the images of the time series, these treelines may easily be located.

A connector may be selected in different ways:

- Select the treeline (see Subsection 4.4.1) and then use `Ctrl+g` while the mouse is pointing to any node of the treeline.
- Use the select tool and click at the base node of one treeline of the connector.
  A dialogue will ask you which object to select. Note that you may control if connectors may or may not be selected using the select tool in the Color&Visibility tab.

- Click on the connector in the `Z space` tab.

If the base nodes of treelines attached to one connector have different geometric positions, e.g. due to non-perfect alignment between images or root position shift, there are color cues indicating the position shift of connector nodes relative to the nodes in the next and/or previous layer (see Figure 14).

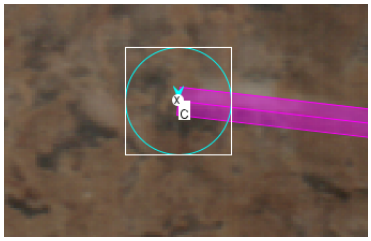
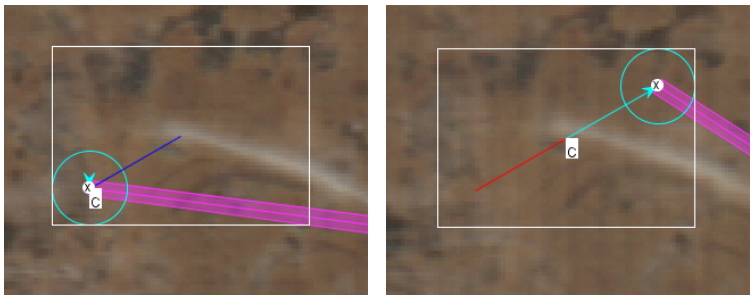


Figure 13: Display of the connector nodes at the base node of a treeline within the connector.



(a) (b)

Figure 14: Example of color cues indicating the position shift of end nodes relative to the origin node of a connector. Left: The blue lines next to the origin node indicate the relative position of the base node of the treeline in the next layer. Right: The red lines indicate the relative position of the current end node to the origin node in the previous layer.

In *rhizoTrak*, connectors are automatically created when treelines in the first image of a time series are copied as an initial annotation to the second image. From the second image on, the existing connectors are extended automatically when treelines are copied to the next image (i.e. each connector has one more attached treeline and end node) and new connectors are generated for newly added treelines in the current image.

It is also possible to create or modify a connector manually as described in the following. Modifying connectors is required, e.g., for resolving conflicts due to merging of treelines (see Subsection 5.2).

Hint As manipulating connectors is involved it is recommended to save the project before doing so.

1. To create a new connector, right-click on the entry `rootstack` in the middle column of the main project window and select

   `Add... -> new connector`

2. Click anywhere in the image where a member treeline is located. This will generate an origin node ("o"). The location of this node is irrelevant as it will be changed as soon as the first treeline is added to the connector in the next step.

3. Select the Connector/Merge Tool  located in the *rhizoTrak* main window. Click on each treeline you want to be linked by the connector. The first node from step 2. above will "jump" to the base node of the first treeline you add. Note that you need to click on a node or at least very close to one of the corresponding treeline.

Similarly you may delete a link by clicking on a treeline that is already linked to the connector. This again requires the connector to be selected and the Connector/Merge Tool being active.

4.5 Extracting Experimental Data

The `'Write csv'` button in the `'RhizoTrak Operations'` tab will allow you to write experimental data of your annotated roots to a csv file. In the output options dialogue you may choose

- to generate the data for each segment or aggregated data for each status within a layer,
- experimental data for all layers or the current layer only,
- the unit of measurement,
- a column separator.

In case the experimental data are requested in metric units, i.e., *mm* or *inches*, calibration information needs to be available in the given images. This information may be added, e.g., by the scanning software. If this information is not available, the user will be notified and may choose pixels as unit or cancel the operation. You may configure via

`RhizoTrak Operations -> Preferences -> Configuration`

whether the calibration information found in the image meta information should be shown to you in a message window. This allows you to validate the calibration information, as it may get corrupted by previous operations on the image. Furthermore, only square pixels are supported for metric units.

Detailed Mode Experimental data may be generated for each segment individually as this allows maximal flexibility for subsequent analysis. A segment is uniquely identified by its **rootID**, **layerID**, and **segmentID**. If a treeline is element of a connector, the **rootID** of its segments will equal the connector index in *rhizoTrak*, otherwise they are set to the index of the treeline. Specifically all segments sharing the same **rootID** and **layerID** constitute one treeline, i.e. one root in one of the images. All segments sharing the same **rootID** belong to the same physical root. See Figure 20 for an example csv file. The file generated for detailed experimental data includes the following columns:

experiment The name of the experiment, as given by the respective image names. Available if image names adhere to naming conventions (see Appendix B), otherwise not available (NA).

tube The tube identifier, as given by the respective image names. Available if image names adhere to naming conventions.

timepoint The time point, as given by the respective image names. Available if image names adhere to naming conventions.

date The date, as given by the respective image names. Available if image names adhere to naming conventions.

rootID The ID of the corresponding root of the segment. Roots linked to the same connector will have the same ID across all layers.

layerID The ID of the image, i.e. the index of the layer, starting from one.

segmentID The ID of the segment. This ID is unique within the same treeline.

length The length of the segment in the specified unit.

startDiameter The diameter of the start node of the segment.

endDiameter The diameter of the end node of the segment.

surfaceArea An estimate of the surface area of the segment, assuming that the 3D shape of each root segment is a truncated cone.

volume An estimate of the volume of the segment, assuming that the 3D shape of each root segment is a truncated cone.

children The number of segments that are children of this segment.

status The status label of the segment as an integer (see Subsection 5.3).

statusName The status label of the segment as a string.

Aggregated Mode *rhizoTrak* features to write aggregated experimental data to a csv file. In this case, the length, surface area, and volume of all segments with an identical status label and within the same layer is added up. One line for each of the defined status labels and each layer is generated comprising the following columns. Again the user may choose to write experimental data for the current or all layers.

experiment The name of the experiment, as given by the respective image names. Available if image names adhere to naming conventions (see Appendix B), otherwise not available (NA).

tube The tube identifier, as given by the respective image names. Available if image names adhere to naming conventions.

timepoint The time point, as given by the respective image names. Available if image names adhere to naming conventions.

date The date, as given by the respective image names. Available if image names adhere to naming conventions.

layerID The ID of the image, i.e. the index of the layer, starting from one.

length The added lengths of all segments within this layer and the status label in the last column.

surfaceArea The sum of the estimated surface area of all these segments with the same geometric assumptions as for detailed mode.

volume The sum of the estimated of the volume of all these segment with the same geometric assumptions as for detailed mode.

status The status label of the segments as an integer (see Subsection 5.3).

statusName The status label of the segments as a string.

ImageWidth The width of the image, NA if not image as been imported to the layer

ImageHeight The height of the image, NA if not image as been imported to the layer

Segments outside the image extend If a segment is completely outside the image extent it is ignored, i.e. omitted, in the output. If a segment is partially outside only the part of the segment within image extents is counted, i.e. the segment is truncated when computing the output. In both cases the annotation itself is left untouched. With respect to the affiliation of segments to treelines *rhizoTrak* preserve the root structure, i.e. all segments are interpreted to belong to the same single treeline, also if they are disconnected due to annotated segments outside the image extend. In the example all four segments completely or partially within the

images (s1, s2, s4, and s5) are counted as belonging to the same treeline. The segments s1 and s5 are completely included in the experimental data file, while segments s2 and s4 are only partially included.

5 Advanced Topics

5.1 Open and Manage an Existing Project

Each *rhizoTrak* project has two associated files which are stored in the project directory. It is the directory chosen upon its creation:

1. `<projectname>.xml`

   the main project file. Depending on the size, this main project file may have been compressed and is named `<projectname>.xml.gz` in this case.

2. `<projectname>.rtk`

   contains *rhizoTrak* specific configurations of the project, e.g. define segment status labels and the directory where to search for new images, and the `connectors` to identify identical roots in different images, i.e. time points.

rhizoTrak also stores some information in a directory named `trakem2....`, e.g. `trakem2.1520595824773.1026709460.106854841` used internally by *trakEM2* and *rhizoTrak*.

Note, that the project name is not only defined by the file names, but also within `<projectname>.xml`. Thus renaming the project files will also require editing of `<projectname>.xml` for consistent naming.

In addition there is one user specific configuration file `settings.xml` which is valid for all *rhizoTrak* sessions of the user. When closing and saving a project the current settings as set by the user via `RhizoTrak Operations -> Preferences` are saved to this file. The user specific configuration subsumes the defined status labels (however not the assignment to status label integers) and their abbreviation, color, alpha value, and selectability (see Subsection 5.3), and further settings (see Subsection 5.4). This configuration is stored in the directory `.rhizoTrakSettings` which is located in the user's home directory, depending on the operating system:

Windows: The user directory of the currently logged in user, e.g., `C:\Users\someuser\.rhizoTrakSettings\settings.xml`

Linux: The user home, e.g., `/home/someuser/.rhizoTrakSettings/settings.xml`

Mac: The user home, e.g., `/Users/someuser/.rhizoTrakSettings/settings.xml`

Technically user's home is defined as the property `user.home` of the Java virtual machine.

To open an existing *rhizoTrak* project run the *rhizoTrak* plugin from Fiji

`Plugins -> rhizoTrak-> Open project`

and open the main project file `<projectname>.xml`.

Additional images may be added in the same way as the first import described in Subsection 4.3. This additional images are always inserted as subsequent time points to the time series already imported.

5.2 Handling Connectors in Merging and Splitting Treelines

Connectors in *rhizoTrak* are used to explicitly represent the identity of a root visible in different images of a time series. The basic principle for using connectors in *rhizoTrak* is to represent a physical root in each layer by one single treeline, and to link all projections of the same physical root in different layers of a time series with a single connector. This implies that the following rules need to hold to yield a consistent annotation:

Rule 1:
Each connector can only include a single treeline from one layer, i.e. no connectors can be added between different treelines in the same layer.

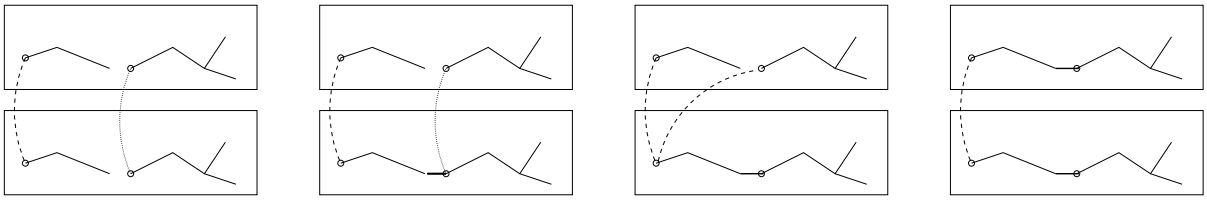
Rule 2:
Each treeline may only be part of one connector.

Usually connectors are created by copying treelines between layers (cf. Section 4.4.2) and most of the time there is no need for the user to manipulate connectors manually. However, merging treelines may lead to violation against rules for connectors and require manual correction of connectors.

There are two types of situations in which manual manipulation of connectors is necessary. The first is a **Multiple Connector Conflict** where rule 2 is violated, i.e., a single treeline is linked to two connectors. The second is a **Multiple Treeline Conflict** where rule 1 is violated, i.e., two treelines are attached to one connector. These two conflicts may occur at the same time.

A typical scenario giving rise to this conflict is as follows (see Figure 15). Copying treelines from one layer to the next results in connectors being automatically added to link each treeline with its copy (Figure 15(a)). If subsequently the two treelines in the second layer are merged, the new treeline is part of two connectors (Figure 15(b)) and a conflict will arise. To inform the user of this situation *rhizoTrak* will pop-up a window and ask if merging should proceed nevertheless. If the user request to proceed the treelines are merged and additionally also the two connectors are "collapsed". Collapsing results in all treelines connect by these two connectors to become members of a newly created connector, while the two old ones are discarded.

In general there are two cases to be distinguished depending on the outcome of "collapsing" two connectors.



(a) before merge (b) after merge (c) collapsed connectors (d) resolved

Figure 15: Example for a merge conflict which is not auto-resolvable. Treelines are depicted with solid lines in two layers. In subfigures (a) and (b) there are two different connectors. They are shown with lines connecting the treelines they contain, which are two treelines in both cases. The connector segments are displayed with the dashed and dotted lines, respectively. In subfigures (c) and (d) the connectors have been collapsed into one single connector, which first contains three, and finally two treelines. It is shown with a dashed line.

1. non auto-resolvable

   This type arises in the example above. Collapsing will break rule 1 as the collapsed connector includes two different treelines from layer 1 (Figure 15(c)). This is called a **Multiple Treeline Conflict**. In the example exactly one conflict will arise. If however treelines from several layers are part of the collapsed connectors, several conflicts will results.

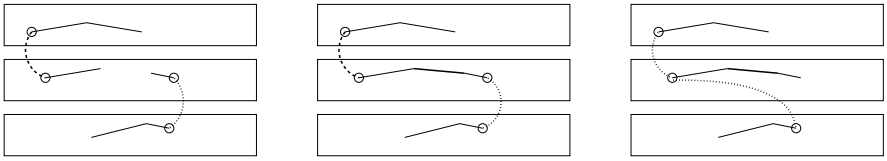
   If the treelines in the first layer are also merged (see Figure 15(d)) then the situation is consistent again. A second way to resolve the conflict would be to remove one of the treelines in the first layer from the connector, which however in this example would not be sensible.

   The conflict manager described below supports the resolving of conflicts.

2. auto-resolvable

   An example for this type is given in Figure 16. Only in the layer where merging is performed, there is one treeline from each of the two connectors, which are depicted as dashed resp. dotted lines in the figure, see Figure 16(b). All other treelines belonging to one of these connectors are in their own layer. Thus collapsing the connectors does not result in a Multiple Connector Conflict (see Figure 16(c)).

   On merging, *rhizoTrak* again informs the user that this would result in a Multiple Connector Conflict which is in this case auto resolvable. If the user proceeds, he or she is asked whether to collapse or not to collapse the connector. If the user decided against collapsing a Multiple Connector conflict is reported in the conflict manager.



(a) before merge (b) after merge (c) collapsed connectors

Figure 16: Example for an auto-resolvable merge conflict. Treelines in three layers are depicted with solid lines. The two connectors in (a) and (b) are shown with a dashed and dotted line, respectively. They have been collapsed into one single connector in subfigure (c).

Conflict Manager *rhizoTrak* features a conflict manager in the `'RhizoTrak Operations -> Conflicts'` tab to support handling connector inconsistencies. The manager will list all current inconsistencies and conflicts, and provides means for resolving them. Note that inconsistencies and conflicts should by all means be resolved as soon as they emerge. Ignoring them and continuing with editing operations on treelines might lead to increasingly complex constellations becoming more and more difficult to resolve.

Hint As conflict managing is involved it is recommended to save the project in advance.

The conflict manager can be opened via the `RhizoTrak Operations -> Conflicts`. A window will pop up and display detected conflicts. The user can select one of the conflicts and click on it.

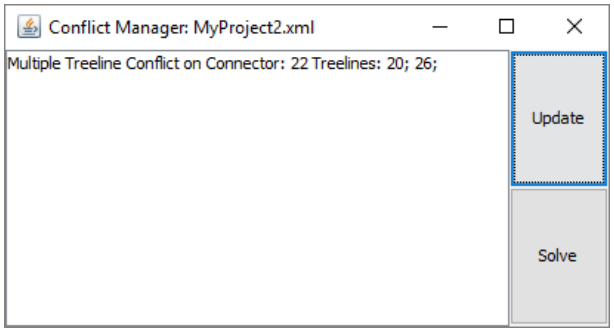


Figure 17: Window of the Conflict Manager.

Multiple Connector Conflict Depending on the situation there are two options

1. In case collapsing the connectors does not result in Multiple Treeline Conflicts, as in the second example above (see Figure 16), clicking the `Solve` button will ask the user if "auto resolve" should be conducted.
2. If collapsing provokes Multiple Treeline Conflicts as in the first example from above (see 15), after clicking the `Solve` button the user is informed of this fact and asked if collapsing should proceed. If so, one or more Multiple Treeline Conflicts will occur and are displayed in the conflict manager.

Multiple Treeline Conflict These need to be resolved manually by merging treelines or removing connectors. This process is supported by the Conflict Manager via the `Solve` button. The treelines involved in the conflict are highlighted in the layer window and the display is switched to the corresponding layer.

While solving conflict, the "highlight color 2" may be used, in cases where high lighting is required which is to be distinguished from already high lighted treelines.

5.3 Manage Segment Status Label and Display Colors

rhizoTrak supports user-defined status labels for each segment. As mentioned in Subsection 4.1 the labels may be provided during creation of the project, otherwise the set of default labels LIVING, DEAD, DECAYED, and GAP is provided. In addition there are four internally used labels, namely UNDEFINED (see below), CONNECTOR (which is used to display the special treelines of connectors), VIRTUAL and VIRTUAL_RSML (see Subsection 5.5).
The segment status labels can be later on changed via using

`RhizoTrak Operations -> Preferences`

as described below. Advanced user may also edit the project specific configuration file `<projectname>.rtk`. The format of this file is described in Subsection C. In this case modifications will take effect when the project is reopened.

It is important to understand the implementation of status labels. Internally the status is represented as an integer value. At any time of the existence of a project a contiguous range of integers starting with zero is available as a valid status. The size of this interval may be changed by the user, see below. The meaning of a segment status integer is defined by an assignment of status label names to these integer values. Modifying this assignment does not change the integer values stored at each segment, rather the interpretation of the status. The current assignment may be displayed and modified via

`RhizoTrak Operations -> Preferences -> Label Mappings`.

The first name and abbreviation listed is assigned to all segments with status 0 (see Figure 18). When opening the project this is the first name listed in the project specific configuration. The next name is assigned to all segments with status 1 and so forth. If for example the name of the second status label is changed all segments with current status 1 will change their meaning to the new name. Furthermore, the number of assigned status labels may be decreased with the `Remove Mapping` button, which will always remove the last name. This will render all segments with the corresponding integer status as UNDEFINED while the integer value of the segments is left untouched. Thus adding a name using the `Add Mapping` will assign the new name to all these segments. If the user adds a name it will always appended at the end of the list. When adding a name or changing a name, the user may always choose from the set of currently defined status labels which is described in the next paragraph. It is valid to assign the same status label name to several integer status values.

Note Internally used status labels, e.g., UNDEFINED, VIRTUAL, are encoded with negative integers and may not be modified by the user.

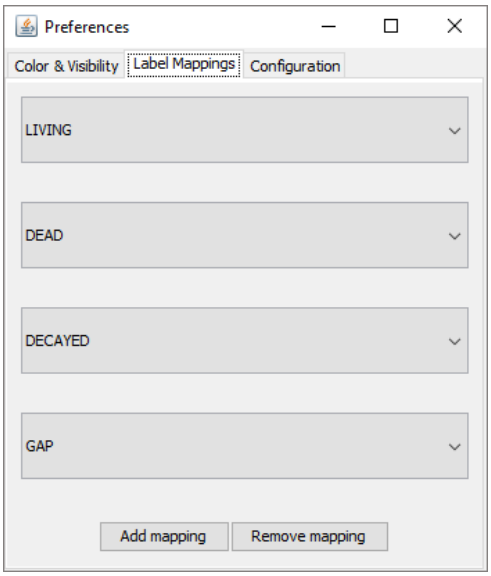


Figure 18: The assignment of integer status labels is displayed in this panel and may also be modified.

The set of defined status label may be inspected in the Color&Visibility panel which may be accessed via

`RhizoTrak Operations -> Preferences -> Color & Visibility`

(see Figure 19). In this window, you can add and delete status labels and modify the abbreviation. The color used to display segments with this status may be changed by clicking on the colored rectangle on the right. The visibility and transparency may be modified by dragging the `alpha` slider. An alpha value of zero is equivalent to invisible. Here you may also choose whether or not treelines with this status are selectable (see below).

Furthermore, the color for highlighting (see Subsection 4.4.3 and 5.2) and the color of the active node of a treeline may be adjusted.

These settings shown and configurable in this Color & Visibility panel are independent of your project and will be saved in your user data or home directory on your local computer whenever you save a project and the user confirms (see Subsection 5.1).

5.4 User configuration

Besides these options to configure status labels, colors, and selectability the user can configure several other properties via

`RhizoTrak Operations -> Preferences -> Configuration`

Figure 19: The Color & Visibility panel with the default status labels LIVING, DEAD, DECAYED, and GAP.

These are

- Confirm merge treelines
  When merging treelines the user is always asked to confirm
- Confirm split treelines
  When splitting treelines the user is always asked to confirm
- Full GUI
  The graphical user interface includes all features of TrakEM instead those required for root annotation. **Note:** If the setting is changed, it takes effect upon restart of *rhizoTrak*.
- Show calibration info
  When writing experimental data with metric units read from the calibration information in the images, this calibration information read is displayed to the user.
- Diameter as circles
  If enabled the diameter of the treeline nodes is displayed as a circle.
- Show diameter lines
  If enabled the diameter of the treeline nodes is displayed as a line perpendicular to the segment.
- Show borders of segments
  If enabled the left and right border of each segment is drawn.
- Fill segments
  The segments are filled with the color and alpha value of the status label of the segment.

As the settings of colors and selectability, these settings are saved into the user specific configuration file, see Subsection 5.1.

5.5 Import and Export of RSML Data

rhizoTrak supports import and export of root annotation data in the Root System Markup Language (RSML). RSML has been devised to yield a common fundament to represent data about roots and root systems ([3]). This facilitates data exchange with other tools featuring RSML support, e.g., RSML adapters for Excel and R are provided for further data processing and analysis.

In RSML the annotation of each image is stored in a separate file. An image is annotated as a scene with a set of plants each containing one or more root systems. A root system is represented as a hierarchically ordered set of non-branching roots. The geometry of a non-branching root is represented with a polyline and may be augmented by, e.g., diameter or status labels using so-called functions. Additional information may be added as properties or annotations on the scene, plant, or root level.

One or multiple RSML files can be selected and imported into *rhizoTrak* using

`RhizoTrak Operations -> ReadRSML`

in a way similar to loading images. If multiple RSML files are loaded they a loaded into consecutive layers. The user needs to specifiy the first layer which may already exist or be a new layer appended at the end of currently existing layers. An RSML file may be imported into a new layer to be created or into an empty (first) layer. In this case the image specified in the RSML file is loaded into the layer. An RSML file can also be imported into a layer with an image already loaded. In this case rhizoTrak validates that this image and the one specified in the RSML file agree using their SHA-256 codes. If inconsistent, the user may choose to cancel the import operation or proceed.

An RSML file may define a mapping of status label integers to string representations in the property list of the scene (see D for details). If a mapping defined in an RSML file contradicts the mappings currently defined in the project or is considered an error, the user is asked, if the already existing mapping should be used and errornous mappings should ne ignored, or if import of the RSML file(s) is to be aborted. The current mapping of the project may be extended with additional mappings in the RSML file. Multiple RSML files are processed in their temporal order.

For each top level root and all its child roots one single *rhizoTrak* treeline is created. As suggested by the RSML thesaurus[5], rhizoTrak uses the `parent-node` property of an RSML root to define the connecting node in the parent root of higher order in the hierarchy. If the `parent-node` property is not given the nearest node in the parent root is used. The status label of the base node of the child root distinguishes both cases, using VIRTUAL if defined via the `parent-node` property, VIRTUAL_RSML otherwise. In addition to the polyline geometry, diameter and status label integers of nodes are imported if available in the RSML files as the `diameter` resp.

[5] `http://rootsystemml.github.io/format`, accessed Oct 11 2018

`statusLabel` function. If no status labels are present in the RSML file, a user defined default is used.

Since rhizoTrak features no concept of plants with multiple root systems, this information is not available to rhizoTrak users, however, preserved on subsequent export.

After importing all RSML files, rhizoTrak checks if all imported RSML files indicate that temporal links are represented. This is the case if all `unified` flags in the time series meta data are true in the RSML files. In this case connectors are established between treelines accordingly.

The current or all layers may be exported to RSML file(s) via

`RhizoTrak Operations -> WriteRSML` ,

where each layer is exported into one RSML file. If all layers are to be exported, the single filename selected by the user is extended with the timepoint of each layer. The status label mapping of the current project is saved in the property list of the scene in each RSML file.

If a treeline has been previously imported its membership to a plant is preserved. Each other treeline is exported in one newly created plant. Each treeline is exported as a hierarchy of polylines accoording to RSML specifications. If the branches have been created by user actions within rhizoTrak or have been read from an RSML file using the `parent-node` property they are saved with RSML's optional `parent-node` property. In addition the diameters of nodes and status labels of segments are stored as RSML functions. If the geometry of a previously imported treeline has not changed after import, additional features like RSML functions and annotations are re-saved as required by RSML specifications.

The user may choose if temporal links are to be represented on export which is accomplished setting the `unified` flags in all RSML files to true and generating unique `rootID`s for the toplevel polylines of all treelines which are members of the same connector.

5.6 Project Structure

This subsection sketches the project structure which is the basis for annotation in *trakEM2* and thus *rhizoTrak*. For most cases the default rootstack structure should be sufficient, and this subsection might be skipped by most users.

Image annotation in *trakEM2* builds upon an hierarchical treelike structure, which allows for assembling annotated entities into semantically meaningful, hierarchical groups. *rhizoTrak* uses treelines and connectors as these annotation entities. In case of neuronal circuits this might for example result in a hierarchy starting with the brain as its root entity and being composed of neurites, synapses, glia and other structural parts. Each hierarchy can be individually structured, e.g., according to semantic or topological criteria of the specific application domain to be modeled. Besides reflecting semantic structures the groups also add to a better usability of the software in case of large amounts of annotated objects, e.g., by allowing for flexible configuration of visualization options for individual groups.

rhizoTrak uses a shallow hierarchy and collects all treelines which annotate root systems and connectors representing the identity of physical roots in different time points below a `rootstack`:

```
rootstack
├─ treeline
└─ connector
```

To allow as much flexibility as possible, further treelines and connectors may be used within a *rhizoTrak* project to annotate semantically different entities in an image. The only requirement by *rhizoTrak* is, that these treelines and connectors are not located below a rootstack in the project structure. For completeness it is mentioned that *rhizoTrak* also accepts more then one rootstack in the project structure and takes into account all treelines below a rootstack as roots, e.g., when computing a experimental data to write to a csv file. However, it is not recommended to use such a project structure.

References

[1] Albert Cardona, Stephan Saalfeld, Johannes Schindelin, Ignacio Arganda-Carreras, Stephan Preibisch, Mark Longair, Pavel Tomancak, Volker Hartenstein, and Rodney J Douglas. Trakem2 software for neural circuit reconstruction. *PloS one*, 7(6):e38011, 2012.

[2] Amram Eshel and Tom Beeckman. *Plant roots: the hidden half*. CRC press, 2013.

[3] Guillaume Lobet, Michael P Pound, Julien Diener, Christophe Pradal, Xavier Draye, Christophe Godin, Mathieu Javaux, Daniel Leitner, Félicien Meunier, Philippe Nacry, et al. Root system markup language: toward an unified root architecture description language. *Plant Physiology*, pages pp–114, 2015.

[4] J Schindelin, I Arganda-Carreras, et al. Fiji: an open-source platform for biological-image analysis. *Nature methods*, 9(7):676, 2012.

A Command reference

Note that this page only lists shortcuts that are relevant to rhizoTrak (e.g. 'displayables' refers only to treelines and connectors). SW = scroll wheel.

A.1 Universal shortcuts

Mouse and key combinations	**Function**
Ctrl + a	Select all visible
a	Create new treeline
Ctrl + z	Undo
Ctrl + Shift + z	Redo
Ctrl + SW **or** '+', '-'	Zoom
SW **or** ',', '.' **or** '<', '>'	Scroll through layers
Shift + Click	Deselect
ESC	Deselect all
Shift + f	Hide/unhide node tags
Alt + f	Hide/unhide treeline arrowheads
q	Hide/unhide all treelines in the current layer
g	Select node (while hovering over it)
Middle Click + Drag	Navigate through layer with hand tool
F4	Text tool
F5	Zoom tool
F6	Hand tool
F7	Line tool
F8	Connector tool
F9	Select tool
F10	Pencil tool
F11	Pen tool

A.2 With displayable or image selected

Mouse and key combinations	Function
j	Open contrast adjustment (images only)
m	Open measurement table (displayables only)
f	Hide/unhide status labels (displayables only)
h	Hide selected
Shift + h	Hide deselected
Alt + Shift + h	Hide deselected except images
Alt + h	Unhide all
Home	Move to top in Z space (displayables)
	or patches (images) panel
End	Move to bottom in Z space (displayables)
	or patches (images) panel
Page up	Move up in Z space (displayables)
	or patches (images) panel
Page down	Move down in Z space (displayables)
	or patches (images) panel
Ctrl + d	Open duplicate in Fiji window (images only)
Del	Delete selected

A.3 With node selected

Mouse and key combinations	Function
t	Add tag to node
Shift + t	Remove tag from node
o	Open diameter adjustment
b	Go to previous branch node or start
n	Go to next branch node or end
e	Go to last edited node
l	Go to last added node
r	Go to base node
d	reroot, i.e. make the selected node the new base node ot the treeline
Shift + SW	Change status label
Shift + Ctrl + Alt + SW	Change status label of subtree
Shift + Alt + SW/Drag	Adjust diameter
Shift + Click	Add node on an edge
Shift + Ctrl + Click	Delete a node
Shift + Ctrl + Alt + Click	Delete a node and its entire subtree

A.4 Additional shortcuts when using the full GUI

Mouse and key combinations	Function
Ctrl + f	Search
Alt + Shift + i	Import image into current layer
Alt + i	Import next image in image directory into current layer
t	Enter affine transform mode
Shift + t	Enter non-linear transform mode
Enter	*Apply (in transform mode)*
ESC	*Cancel (in transform mode)*
Shift + c	Open color adjustment

B Image naming conventions

The following filename convention is assumed for the sorting function when importing images, and for some columns of the experimental data written to csv files. It follows the ICAP naming convention with slight relaxations. This conventions is attributed to Bartz Technology Corporation in [2].

`experiment_T<N>_imagetype_date_timeofday_timepoint_experimenter.extension`

`experiment` a string which identifies this experiment uniquely and distinguishes it from other (minirhizotron) experiments

`T<N>` a string identifying a tube uniquely within this experiment where `<N>` is the number optionally with leading zeros.

`imagetype` describes the various images which are associated with one tube (in one experiment) for a given scan, e.g. scans in different depth of a tube

`date` the day when the original image was scanned

`timeofday` the time of day when the original image was scanned

`timepoint` a integer value – optionally with leading zeros – which gives the sequential number of the scan, i.e. the time points, typically starting with 1.

`experimenter` a string to identify the experimenter(s) who acquired the image

For sorting only the `timepoint` is used, all other parts are ignored.

C File formats

C.1 Project Configuration File

The format of project configuration file `<projectname>.rtk` for the default status labels is as shown in the following example. The names and abbreviations are mapped to integer values defined for each segment as the status in the order they are listed in the file, starting from 0 (see Subsection 5.3).

It is possible to supply a user defined configuration file during creation of a new project where the attribute `fullName` and `abbreviation` need to be edited.

C.2 Output of Experimental Data

An example of a output file for detailed experimental data is given in Figure 20, for the aggregated mode in Figure 21.

D Details for RSML

The definition of status label mappings is expected and generated as follows: In the property definitions in the meta data section the property `StatusLabelMapping` as a integer-string pair is defined:

```
<property-definition>
    <label>StatusLabelMapping</label>
    <type>Integer-String-Pair</type>
</property-definition>
```

The actual definiton of mappings is given for each defined status label as one property element with attributes `int` and `value` in the property list of the scene, e.g.:

```
<statusLabelMapping int="0" value="LIVING"/>
<statusLabelMapping int="1" value="DEAD"/>
<statusLabelMapping int="2" value="DECAYED"/>
<statusLabelMapping int="3" value="GAP"/>
```

The mappings need to define a contigous interval of non negative integers including zero. Other mappings, including mappings with negative integer labels, are considered an error.

Several examples on the RSML homepage (http://rootsystemml.github.io/format, accessed Oct 11 2018) and shipped with the RSML conversion tools (https://github.com/

RootSystemML/RSML-conversion-tools, accessed Oct 8 2018) represent the sample values of functions with text content in contrast to attributes as defined in the XSD schema definition. To handle both variants rhizoTrak supports both forms of the XML representations on import.

E Known Issues

1. In case the internal directory `trakem2.1520595824773.1026709460.106854841` (see Subsection 5.1) was deleted or tampered with, a window as shown in Figure 22 pops up when opening the project. However, *rhizoTrak* will still start-up properly.

experiment	tube	timepoint	date	rootID	layerID	segmentID	length_pixel	startDiameter_pixel	endDiameter_pixel	surfaceArea_pixel^2	volume_pixel^3	children	status	statusName
me	T14	1	25.02.15	22	1	1	212.8097742116184	42.66666793823242	42.66666793823242	28525.296776533847	304269.8428622998	1	0	LIVING
me	T14	1	25.02.15	22	1	2	223.5754906066405	42.66666793823242	37.33333206176758	28097.32313266312	281369.476940063	0	2	DECAYED
me	T14	2	12.03.15	22	2	1	212.8097742116184	45.333335876464844	39.33333206176758	28305.255862100996	300035.61216388916	1	0	LIVING
me	T14	2	12.03.15	22	2	2	223.5754906066405	39.33333206176758	40.0	27861.22622563362	276296.69697091135	0	1	DEAD
me	T14	2	12.03.15	28	2	1	436.2499283667563	50.66667175292969	34.666664123535156	58485.33483935011	631048.1623243917	1	0	LIVING
me	T14	2	12.03.15	28	2	2	354.6223343220221	34.666664123535156	34.666664123535156	38621.39973732163	334718.771909407	1	3	GAP
me	T14	2	12.03.15	28	2	3	240.46829312822095	34.666664123535156	0.0	13128.498924184294	75657.25023656912	0	0	LIVING
me	T14	5	23.04.15	28	3	1	281.82441342083905	42.66666793823242	34.666664123535156	34238.04275153295	332114.9333253942	1	0	LIVING
me	T14	5	23.04.15	28	3	2	321.6022387981775	34.666664123535156	26.666664123535156	30986.252969963643	238890.00336150735	0	0	LIVING
me	T14	2	12.03.15	30	2	1	202.40553352119602	42.66666793823242	29.333330154418945	22903.940202846592	208378.82284157196	1	0	LIVING
me	T14	2	12.03.15	30	2	2	144.49913494550754	29.333330154418945	40.0	15747.905034841748	137465.0221781038	0	0	LIVING

Figure 20: Example for a csv file with detailed experimental data.

experiment	tube	timepoint	timepoint	layerID	length_pixel	surfaceArea_pixel^2	volume_pixel^3	status	statusName	ImageWidth	ImageHeight
me	T14	1	25.02.15	1	212.8097742116184	28525.296776533847	304269.8428622998	0	LIVING	5100	710
me	T14	1	25.02.15	1	0.0	0.0	0.0	1	DEAD	5100	710
me	T14	1	25.02.15	1	223.5754906066405	28097.32313266312	281369.476940063	2	DECAYED	5100	710
me	T14	1	25.02.15	1	0.0	0.0	0.0	3	GAP	5100	710
me	T14	2	12.03.15	2	1236.4326641732991	138570.93486332372	1352584.8697445258	0	LIVING	5100	710
me	T14	2	12.03.15	2	223.5754906066405	27861.22622563362	276296.69697091135	1	DEAD	5100	710
me	T14	2	12.03.15	2	0.0	0.0	0.0	2	DECAYED	5100	710
me	T14	2	12.03.15	2	354.6223343220221	38621.39973732163	334718.771909407	3	GAP	5100	710
me	T14	5	23.04.15	3	603.4266522190165	65224.29572149659	571004.9366869015	0	LIVING	9206	710
me	T14	5	23.04.15	3	0.0	0.0	0.0	1	DEAD	9206	710
me	T14	5	23.04.15	3	0.0	0.0	0.0	2	DECAYED	9206	710
me	T14	5	23.04.15	3	0.0	0.0	0.0	3	GAP	9206	710

Figure 21: Example for a csv file with aggregated experimental data.

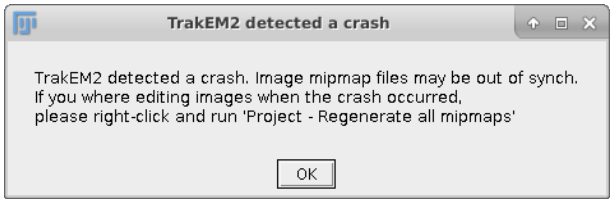


Figure 22: Known issue on opening project with corrupt `trakem2....` directory