Supplementary Information S1: (modified 2020-02-25) PSFtracker Workflow Description and Manual

This document describes the workflow from recording of the subresolution beads and PSFj application to the usage of PSFtracker, with particular emphasis on the latter.

Preconditions for PSFtracker application

Reasonable application of PSFtracker requires that test images of beads were recorded and evaluated with PSFj at least twice, but preferably more often. With a laser scanning microscope images can be generated with gold beads in reflection mode or with fluorescent beads. For other fluorescent microscopes only fluorescent beads are suitable.

For microscope image acquisition, parameters generally should be set according to the PSFj manual (http://www.knoplab.de/psfj/online-manual/).

We recommend to record 3D bead images for two wavelengths and – on confocal systems – with two pinhole settings. Images recorded with fully open pinhole allow visual inspection of the side flares of the PSF. A pinhole setting of e.g. 0.5 allows to determine which resolution can be achieved under optimal imaging conditions. For wavelengths, we use 488 nm and 638 nm, since they are available on all our confocal microscopes. (The 405 laser cannot be used for reflection mode imaging on a Leica SP8).

File format and file name

PSFj needs each image stack provided as one tiff stack. This is easily accomplished by opening the original file from the microscope software in Fiji (Schindelin et al., 2012) where channels can be split and can be saved as tiff.

For any workflow, consistent naming of files is crucial. PSFtracker needs to be able to extract metadata from the filename, therefore the following nomenclature is mandatory:

YYYYMMDD_ System_Objective_Wavelength_pinhole_comment

Example: 20171005_scope1_63xoil_488_05_newoil.tif

KNIME will find the blocks of metadata between the underscores. Do not use additional underscores or additional dots in the file name, i.e. do not give the pinhole size as 0.5 but as 05 instead, etc.

Date - Use the format YYYYMMDD.

System - Name of the microscope, e.g. as used in a booking system.

Objective - Specify a unique identifier for the objective on this system.

Wavelength - Center wavelength for the used emission filter (fluorescence) or excitation wavelength (reflection mode).

Pinhole - Specify size of the pinhole used for the measurements (use e.g. “open” for widefield systems).

Comment - Any comment that may be helpful, e.g. to distinguish a data set from others from the same day.

Application of PSFj

Each bead image file is processed by PSFj as described in the PSFj-manual. Results must be saved as .csv files and **in the same folder as the image stack**. In our hands, not all beads in an image stack are usually recognized by PSFj, sometimes not even half. However a big enough fraction is usually evaluated. In PSFj it is possible to load several images and calculate the PSF-statistics over all beads in all images. Please do not do this, but evaluate every image separately.

We have used PSFj v2.0 build 241. It is possible that newer versions of PSFj change the internal structure of the csv-file. This could potentially cause errors when the result files are processed in KNIME.

Computer Platform

KNIME is available for Windows, Linux and Mac OS X. In principle PSFtracker should run on all these platforms. However, we only could test it under Windows 10.

Installation of PSFtracker

Installation of R and required R-packages

R is a free software package that provides statistical computing and graphics. The graphics capabilities can be used from within the KNIME environment. PSFtracker needs these capabilities to draw plots. Download the R Project from r-project.org and install it. During development, we used version R-3.5.1. Version 3.6.2 also seems to work.

After installation, start R as administrator (required only for package installation) and in the main menu go to Packages>Install package(s). (Do not confuse with ‘load packages’). In the pop-up window, select a CRAN mirror near your location. In the list of packages, select the following:

- Rserve
- ggplot2
- reshape

Under windows, you can select several packages by using the ctrl-key. Click ok for installation.

When PSFtracker is used, R must be running in the background and the command

```
library(Rserve); Rserve(args = "--vanilla")
```

must have been entered in the R-shell. Do not forget to hit enter. Once this command is executed, no further user operation is needed within the R software,

Installation of KNIME and PSFTracker

- Install KNIME Analytics Platform from www.knime.org. During development, we used KNIME version 3.5.0 and 3.5.1 64 bit. Whether or not PSFtracker will run under future versions cannot be foreseen. 4.1.1 64 bit seems to work fine.
- PSFtracker is provided as a “KNIME workflow file”, file extension knwf, which can be downloaded from our website. After installation of KNIME, start it and in the main

menu go to File > Import KNIME Workflow> Select file: Navigate to PSFTracker.knwf on your local drive to import the workflow. This import is required only once. When you start KNIME next time, the KNIME Explorer (top left window) will already show the PSFtracker.

- Double-click on PSFtracker within the KNIME Explorer (top left window) to load it.
- When PSFtracker is loaded for the first time, a lot of error messages will appear. This is nothing to worry about. Click “YES” to search for missing nodes, then install them by clicking “next” several times in the installation wizard. When asked to restart KNIME do it. Installation is now complete.

For a test run, we provide sample data together with PSFtracker.

How to use PSFtracker

Overview

PSFtracker is a pipeline in KNIME. A KNIME pipeline is generated by adding and connecting so-called nodes. These are displayed in the top central window of the program. A node may have any of a number of functions, like reading data, processing images, analyzing images, etc. It also may contain user defined parameters such as a file path.

PSFtracker reads in the csv-files created by PSFj and processes them as well as the original image stacks. The following files are created:

- Images of one example bead for each evaluated 3D image stack (cropped xy, xz-, yz-views)
- Plots graphically displaying the FWHM over time.
- HTML-File which embeds the above data for better overview and/or for publishing on a website
- XLS-files listing the collected measurements

The Step-by-Step Manual

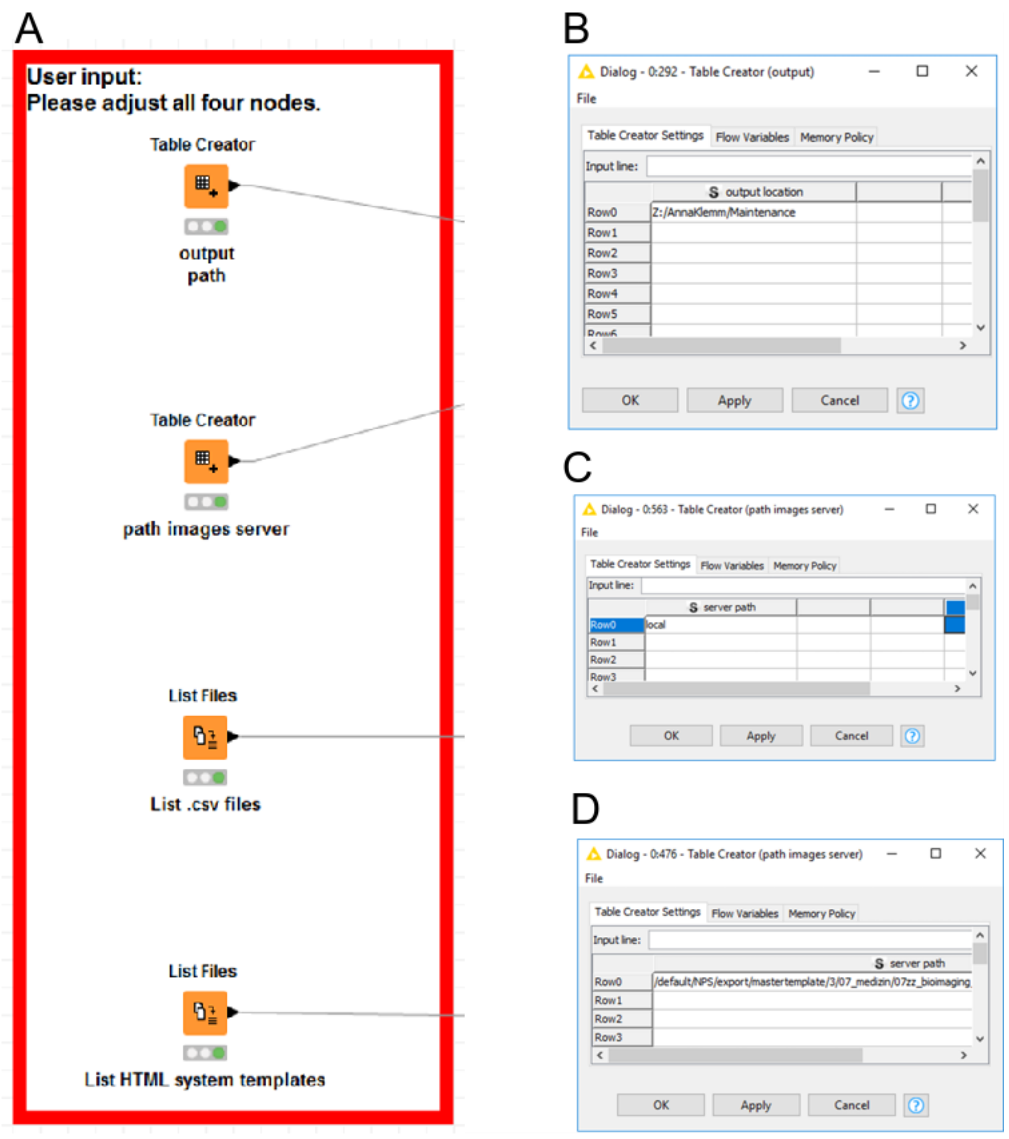
Start R and execute the command

```
library(Rserve); Rserve(args = "--vanilla")
```

Let it run in the background while using PSFtracker.

Start KNIME. When installed as described above, PSFtracker will appear as the last line in the KNIME explorer (top left window). Double click on the name and the PSFtracker nodes will appear in the top central window.

Four nodes (labeled 1 – 4 below) in the top central window need user input. They are all at the left edge of the node map, surrounded by a thick red frame. You may have to use the scroll bar under the window to get to the left edge of the node map. Double-click on a node to enter the required data as follows (see also Figure S1- 1):



Dialog - 0:292 - Table Creator (output)

	S output location
Row0	Z:/AnnaKlemm/Maintenance
Row1	
Row2	
Row3	
Row4	
Row5	

Dialog - 0:563 - Table Creator (path images server)

	S server path
Row0	local
Row1	
Row2	
Row3	

Dialog - 0:476 - Table Creator (path images server)

	S server path
Row0	/default/NPS/export/mastertemplate/3/07_medizin/07zz_bioimaging
Row1	
Row2	
Row3	

Figure S1- 1: A) The four nodes that have to be configured upon start up. B) Table Creator – output path. C) Table Creator – path image server set to ‚local‘ to generate a HTML without aim of publishing the data online, D) Table Creator – path image server with the folder structure in the content manadement system.

1) Table Creator – output path

The output path is the parent folder used for saving all the output-data (Excel sheets, HTML files, plots, images of example beads). KNIME will create several subfolders in the output-folder, where it saves the data output in a structured manner (discussed below). In the first cell of “Row0”, enter the output path in URL format (e.g. D:/Documents/Microscope Maintenance; Figure S1- 1b). Click ok

2) Table Creator – path images server

Double click on the node with that name that is connected to a line at the right black triangle to open it. Enter the correct input (see below) in the first cell of Row0. PSFtracker creates an HTML file that embeds the PSF images that it made. Depending on the selected option, the html file will contain different paths to the files of the PSF images:

a) *Create an HTML-file without the aim of publishing your data online.*
   Enter **local**, as depicted in (Figure S1- 1c). The KNIME pipeline will create a HTML file, which you can use for inspecting your PSF-data, but not for uploading on the web.

We suggest to do a first test run with this option, since the following alternative may be somewhat complicated to set up, depending on your local web site organization.

b) *Create an HTML-file for web publishing with a content management system.*
For the HTML-file to function on the web, the following folder structure in the content management system (CMS) is required:
base-filepath/system/system-objective-psfs/images
where 'system' is the name of a given microscope and system-objective-psfs is a subfolder of which the name must consist of exactly the respective parts to be compatible with the html file generated. For example, the path of the image folder could be base-filepath/scope1/scope1-63x-psfs/images.

Unfortunately, it may be difficult to figure out the required base-filepath within the CMS. It may be different from the URL later used on the web and also from the path displayed within the CMS. One way of figuring out the correct path is to create in the CMS a preliminary HTML page with an embedded image which is located at the respective file system location, then look at the html code of the page to extract the path to this example image.

The base-filepath needs to be entered in Row0 (Figure S1- 1d). Starting from this base-filepath, PSFtracker then builds a HTML-file which includes the path to the images as explained above.

When uploading the PSF images created by PSFtracker to your content management system, make sure the target folder is named exactly according to this scheme.

Hint: A little bit set aside is another node called Table Creator path images server. This node is not connected (with a line) to any other node and thus does not influence PSFtracker results. We use it to permanently store the rather complicated base-filepath from our CMS to easily copy this path into the used Table Creator node described above.

3) List Files – List .csv files
The folder containing the csv-files created by PSFj must be specified using the node ***List Files – List .csv files***. Double click on the node to open it and browse to the folder which contains all .csv files. When the respective box stays checked, all subfolders will be searched for csv files. This way, data for different pinhole sizes and wavelength can be processed together. Click ok. If csv files for different systems and/or different objectives are found, several html files are created.

4) List Files – List HTML system templates
This node allows to customize the output HTML-file by using a template. In the download material is an example HTML-template which can be opened with a text editor or a browser. Our custom template contains introductory information: Which microscope, which objective, and which type of beads was used, which theoretical FWHM is expected. We generated our template in the HTML-editor of our content management system, copied the html code and saved it as a text file. The file extension must be txt, the file name must be according to
html_template_system_objective.txt

Enter the path to the template or templates manually or by browsing to the folder containing them. During browsing, the html files themselves will NOT be visible. An existing folder must be selected, even when no template is used! Click ok.

If the selected folder does not contain templates, PSFtracker will create an HTML-file without customized information. If, however, the folder does contain a template but none that matches the csv files read in, PSFtracker will abort and show an error message in the console (bottom right window).

Program execution

When these four nodes are configured, the pipeline can be executed. R must run in the background (see above).

In KNIME, make sure that the PSFtracker pipeline window (the one with the nodes) is active. Then apply the command “Execute all” in the node menu or use SHIFT+F7. The results are written in the folders as specified above. When PSFtracker is running, several warning messages will appear in the console (bottom right KNIME window). This is no reason for concern. During execution, the currently executed nodes change their traffic light color from red to green. Execution may take several minutes. There is no positive feed back. Your can monitor processor load to see if execution is complete.

The html file is written to a subfolder of the output path parent folder as specified in node 1 above. The subfolder is system\psf_measurements\reports\objective. For example, the file may be under scope1\psf_measurements\reports\63xoil\scope1_63xoil.html

If the local html file option was used, the created file can be directly viewed in a web browser. An example for a local html is included in the software package. Our web site shows an example for an html file embedded in a CMS (http://www.bioimaging.bmc.med.uni-muenchen.de/instrumentation/west-room/malpighi/malpighi-63xoil-psfs/index.html). When KNIME is closed it asks if changes to the work flow should be saved. If you do, your entered paths will be available next time you start KNIME with the PSFtracker.

Trouble shooting

Please note that in the output directories, older files with the same name will be overwritten without asking.

Once PSFtracker has completed you cannot run it again, because all nodes are recognized as completed. To start another run, for example because you added more PSFj output files and image data, right click on the respective node (In this example: node 3, List Files – List .csv files), and select ‘reset’. Only those nodes are run again which are affected by the reseted node. Others will store their previous result. All nodes are reset when KNIME is closed.

As long as all the messages in the console (bottom right KNIME window) start with WARN, they can be ignored. An ERROR however, can lead to abortion of PSFtracker. For example, an error can occur when a required file does not have the correct name or is not available at the correct location.

Detailed description of PSFtracker operating modes

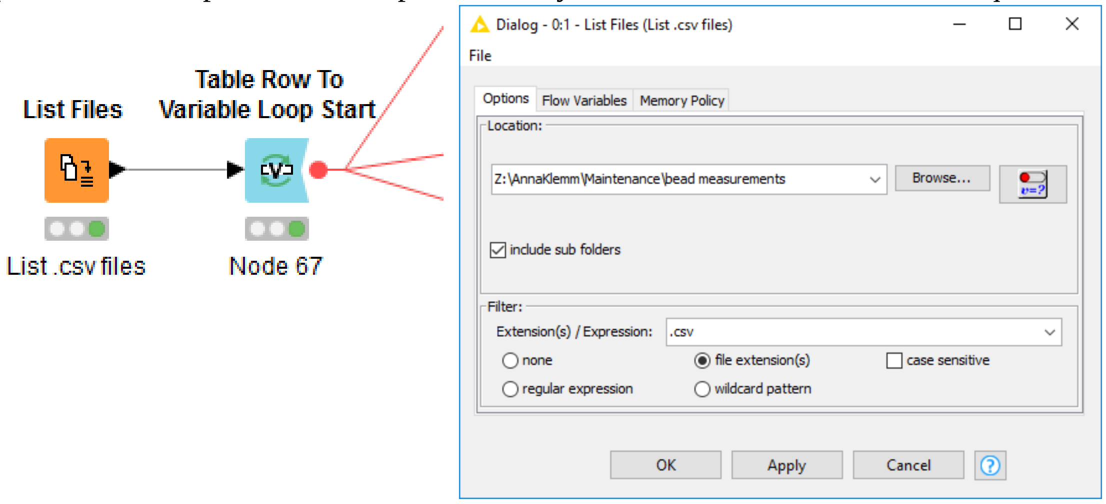
The following sections explain crucial nodes of the PSFtracker pipeline in detail. This should help to modify the software to specific needs, to adapt it if the PSFj file format should change in future versions, or the like. The following is also an illustration on how images and tables can be processed in KNIME that may be helpful in other contexts since descriptions of KNIME pipelines are rare.

An understanding of the following descriptions is, however, not required to run PSFtracker. No adjustments need to be made to any of the other nodes if the software is used as described above.

Reading in the data: List Files, Table Row to Variable Loop Start and File Reader

List Files

List Files lists the path to all csv-files within the folders (+subfolders) selected by the user (see above). This list of the paths pointing to the csv-files is then passed on to ***Table Row to Variable Loop Start***, which passes on the paths one by one to ***File reader*** and subsequent nodes.



Dialog - 0:1 - List Files (List .csv files)

Location: Z:\AnnaKlemm\Maintenance\bead measurements

[✓] include sub folders

Filter:

Extension(s) / Expression: .csv

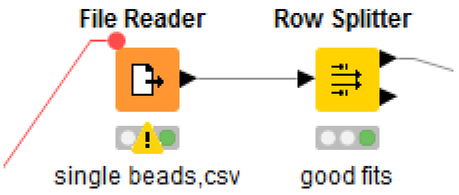
( ) none [● file extension(s)] [ ] case sensitive

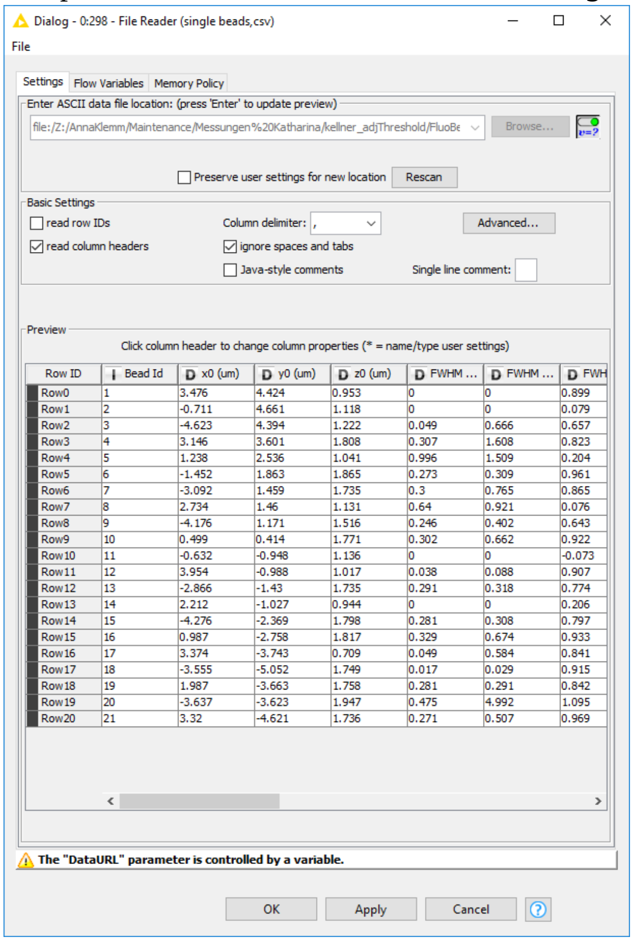
( ) regular expression ( ) wildcard pattern

File Reader

The csv-files are imported using ***File Reader***.

The column headers are read in by checking the respective tick box under ***File Reader/Settings***.




Dialog - 0:298 - File Reader (single beads,csv)

File

Settings | Flow Variables | Memory Policy

Enter ASCII data file location: (press 'Enter' to update preview)

file:/Z:/AnnaKlemm/Maintenance/Messungen%20Katharina/kellner_adjThreshold/FluoBe

Browse...

Preserve user settings for new location ☐ | Rescan

Basic Settings

☐ read row IDs | Column delimiter: , | Advanced...

☑ read column headers | ☑ ignore spaces and tabs

☐ Java-style comments | Single line comment:

Preview

Click column header to change column properties (* = name/type user settings)

Row ID	Bead Id	x0 (um)	y0 (um)	z0 (um)	FWHM ...	FWHM ...	FWH
Row0	1	3.476	4.424	0.953	0	0	0.899
Row1	2	-0.711	4.661	1.118	0	0	0.079
Row2	3	-4.623	4.394	1.222	0.049	0.666	0.657
Row3	4	3.146	3.601	1.808	0.307	1.608	0.823
Row4	5	1.238	2.536	1.041	0.996	1.509	0.204
Row5	6	-1.452	1.863	1.865	0.273	0.309	0.961
Row6	7	-3.092	1.459	1.735	0.3	0.765	0.865
Row7	8	2.734	1.46	1.131	0.64	0.921	0.076
Row8	9	-4.176	1.171	1.516	0.246	0.402	0.643
Row9	10	0.499	0.414	1.771	0.302	0.662	0.922
Row10	11	-0.632	-0.948	1.136	0	0	-0.073
Row11	12	3.954	-0.988	1.017	0.038	0.088	0.907
Row12	13	-2.866	-1.43	1.735	0.291	0.318	0.774
Row13	14	2.212	-1.027	0.944	0	0	0.206
Row14	15	-4.276	-2.369	1.798	0.281	0.308	0.797
Row15	16	0.987	-2.758	1.817	0.329	0.674	0.933
Row16	17	3.374	-3.743	0.709	0.049	0.584	0.841
Row17	18	-3.555	-5.052	1.749	0.017	0.029	0.915
Row18	19	1.987	-3.663	1.758	0.281	0.291	0.842
Row19	20	-3.637	-3.623	1.947	0.475	4.992	1.095
Row20	21	3.32	-4.621	1.736	0.271	0.507	0.969

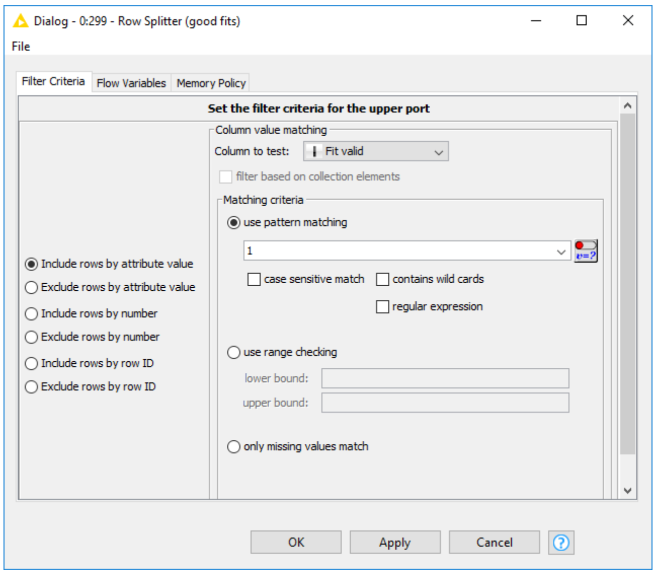
The "DataURL" parameter is controlled by a variable.

OK | Apply | Cancel

Extracting data: Selecting only "good" beads, getting Metadata

Row Splitter

PSFj classifies all fits with an $R^2 > 0.9$ as "good fits" (0 – bad fit, 1 – good fit in column Fit valid). The csv-file contains all beads, also those with bad fits. In KNIME the beads with good fits are filtered using ***Row Splitter***.


Dialog - 0:299 - Row Splitter (good fits)

Set the filter criteria for the upper port

Column value matching
Column to test: Fit valid
filter based on collection elements [ ]

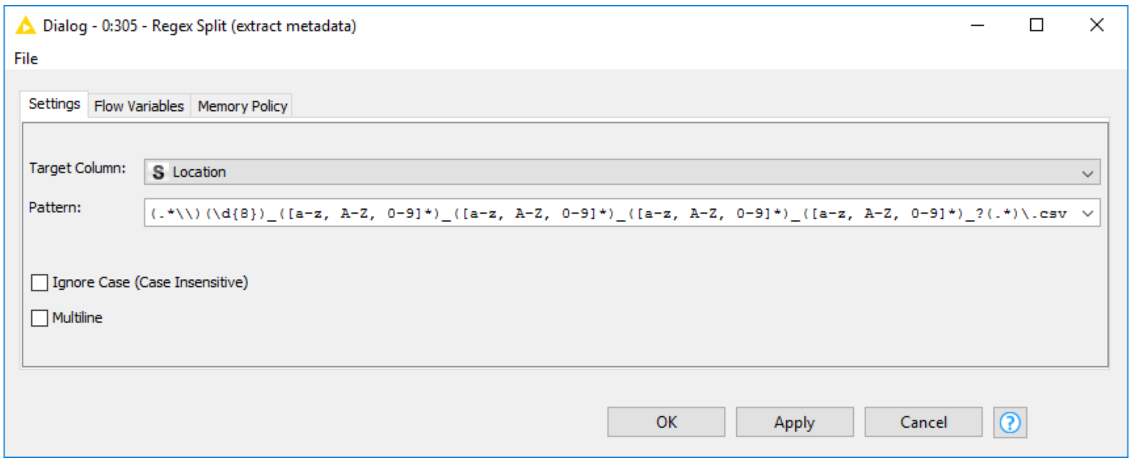
Matching criteria
[✓ use pattern matching]: 1
case sensitive match [ ] contains wild cards [ ] regular expression [ ]
use range checking
lower bound:
upper bound:
only missing values match

[✓ Include rows by attribute value]
Exclude rows by attribute value
Include rows by number
Exclude rows by number
Include rows by row ID
Exclude rows by row ID

Extracting the metadata

From the file name with the structure
YYYYMMDD_System_Objective_Wavelength_pinhole_comment

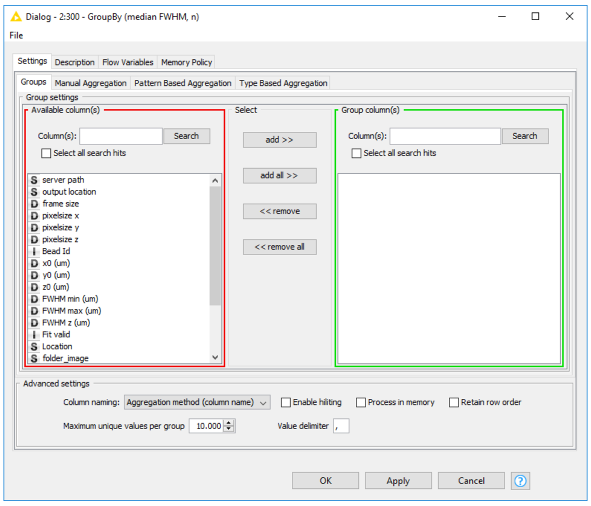
the KNIME-workflow extracts the information using regular expressions inside ***Regex Split***. A good tutorial on regular expressions and an online testing tool can be found at regexr.com. The metadata is then later used for sorting the data into different groups.


Dialog - 0:305 - Regex Split (extract metadata)

Target Column: S Location
Pattern:
```
(.*\\)(\d{8})_([a-z, A-Z, 0-9]*)_([a-z, A-Z, 0-9]*)_([a-z, A-Z, 0-9]*)_([a-z, A-Z, 0-9]*)_?(.*)\.csv
```
Ignore Case (Case Insensitive) [ ]
Multiline [ ]

Calculating bead statistics

To account for asymmetries within the PSF, PSFj calculates the maximal FWHM and the minimal FWHM of the PSF in the focal plane (xy). In addition the FWHM along the z-axis is determined for each bead. PSFtracker then calculates the median values of these FWHMs using ***GroupBy***. ***GroupBy > Groups > Group column(s)*** field was left empty to calculate the median over all beads (beads with $R^2<0.9$ were excluded previously by ***Row Splitter***). We found the median FWHM values to be more reliable compared to the mean FWHM values, since outliers with an extraordinary big FWHM, e.g. caused by bead clusters, can influence the mean.

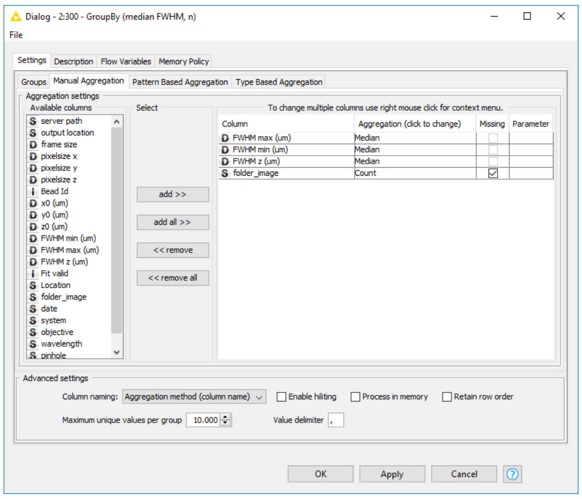

Dialog - 2:300 - GroupBy (median FWHM, n)

Settings > Groups > Group settings

Available column(s): server path, output location, frame size, pixelsize x, pixelsize y, pixelsize z, Bead Id, x0 (um), y0 (um), z0 (um), FWHM min (um), FWHM max (um), FWHM z (um), Fit valid, Location, folder_image

Group column(s): (empty)

Advanced settings: Column naming: Aggregation method (column name); Enable hiliting ☐; Process in memory ☐; Retain row order ☐; Maximum unique values per group: 10.000; Value delimiter: ,


Dialog - 2:300 - GroupBy (median FWHM, n)

Settings > Manual Aggregation > Aggregation settings

Available columns: server path, output location, frame size, pixelsize x, pixelsize y, pixelsize z, Bead Id, x0 (um), y0 (um), z0 (um), FWHM min (um), FWHM max (um), FWHM z (um), Fit valid, Location, folder_image, date, system, objective, wavelength, pinhole

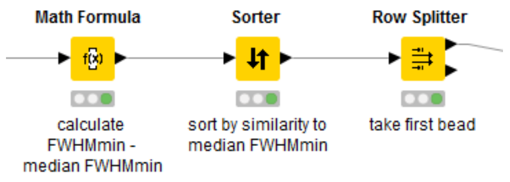
To change multiple columns use right mouse click for context menu.

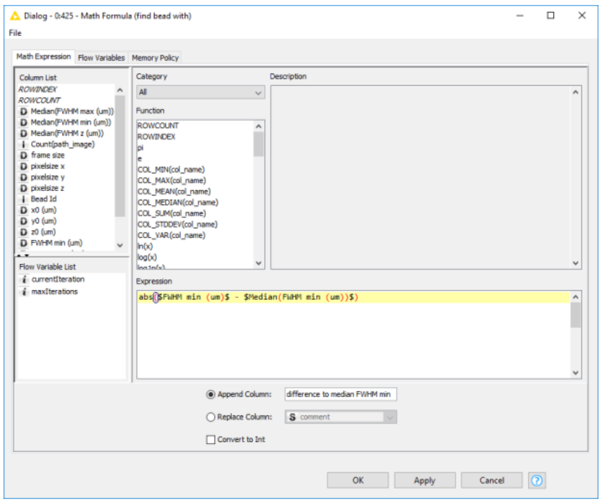
Column	Aggregation (click to change)	Missing	Parameter
FWHM max (um)	Median	☐	
FWHM min (um)	Median	☐	
FWHM z (um)	Median	☐	
folder_image	Count	✓	

Advanced settings: Column naming: Aggregation method (column name); Enable hiliting ☐; Process in memory ☐; Retain row order ☐; Maximum unique values per group: 10.000; Value delimiter: ,

<u>Finding a representative bead for the images</u>

Next to the median FWHM-values, the final table should contain images of one typical bead. PSFtracker selects the bead with the $FWHM_{min}$ closest to the median $FWHM_{min}$-value. For this, first the absolute difference between $FWHM_{min}$ of each bead and the median $FWHM_{min}$ is calculated using ***Math Formula***. Second, the KNIME-table is sorted by ascending difference using ***Sorter***. Third, ***Row Splitter*** selects the first bead in the table – the bead with the smallest absolute difference $FWHM_{min}$ – $FWHM_{min}$-median.


Math Formula
Sorter
Row Splitter
f(x)
calculate FWHMmin - median FWHMmin
sort by similarity to median FWHMmin
take first bead


Dialog - 0:425 - Math Formula (find bead with)

File

Math Expression | Flow Variables | Memory Policy

Column List: ROWINDEX, ROWCOUNT, Median(FWHM max (um)), Median(FWHM min (um)), Median(FWHM z (um)), Count(path_image), frame size, pixelsize x, pixelsize y, pixelsize z, Bead Id, x0 (um), y0 (um), z0 (um), FWHM min (um)

Flow Variable List: currentIteration, maxIterations

Category: All

Function: ROWCOUNT, ROWINDEX, pi, e, COL_MIN(col_name), COL_MAX(col_name), COL_MEAN(col_name), COL_MEDIAN(col_name), COL_SUM(col_name), COL_STDDEV(col_name), COL_VAR(col_name), ln(x), log(x)

Description

Expression:

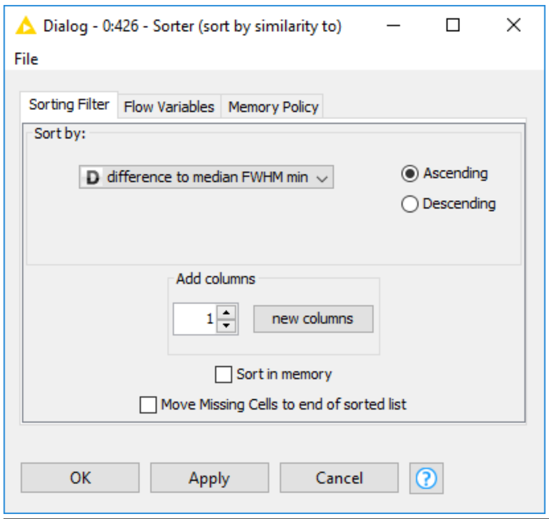
```
abs($FWHM min (um)$ - $Median(FWHM min (um))$)
```

(●) Append Column: difference to median FWHM min

( ) Replace Column: S comment

☐ Convert to Int

OK | Apply | Cancel


Dialog - 0:426 - Sorter (sort by similarity to)

File

Sorting Filter | Flow Variables | Memory Policy

Sort by: D difference to median FWHM min

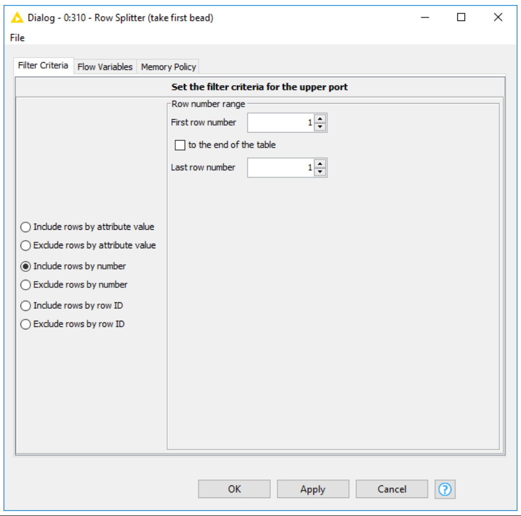
(●) Ascending
( ) Descending

Add columns: 1 new columns

☐ Sort in memory

☐ Move Missing Cells to end of sorted list

OK | Apply | Cancel


Dialog - 0:310 - Row Splitter (take first bead)

File

Filter Criteria | Flow Variables | Memory Policy

Set the filter criteria for the upper port

( ) Include rows by attribute value
( ) Exclude rows by attribute value
(●) Include rows by number
( ) Exclude rows by number
( ) Include rows by row ID
( ) Exclude rows by row ID

Row number range

First row number: 1

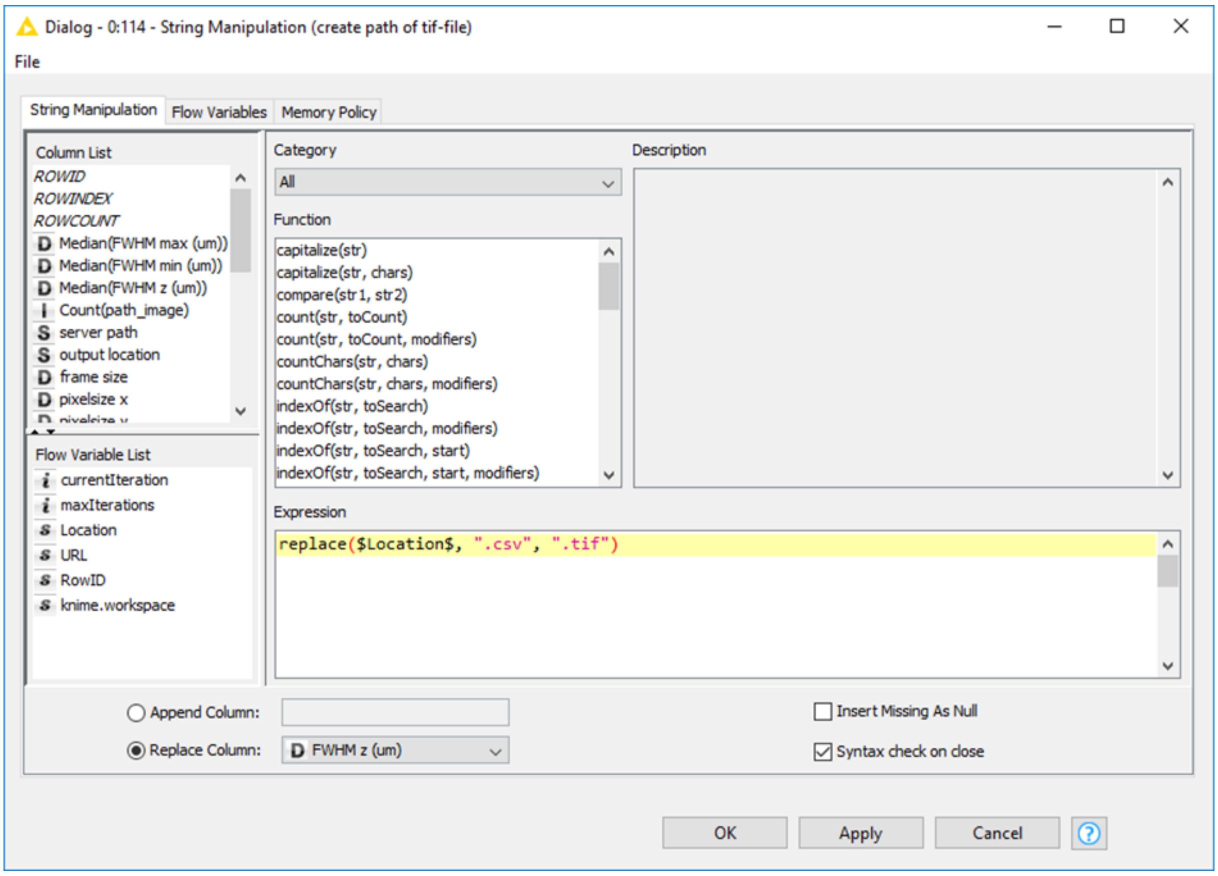
☐ to the end of the table

Last row number: 1

OK | Apply | Cancel

Image Processing in KNIME: generation of xy-, xz-, and yz-views of the representative bead.

A visual presentation of PSF images taken over time is most helpful for continuously assessing the quality of the objective. From the representative bead of each 3D-image (see above) PSFtracker therefore generates cropped images along the xy-, xz-, and yz-plane. Given the location of the original 3D image file, ***Image Reader*** can load it for further processing. PSFtracker expects that the csv-file exported from PSFj was saved into the same folder as the analyzed image. Following this assumption, the image path can be generated by replacing the final ".csv" of the csv-path using ***String Manipulation***.


Dialog - 0:114 - String Manipulation (create path of tif-file)

File

String Manipulation | Flow Variables | Memory Policy

Column List
ROWID
ROWINDEX
ROWCOUNT
D Median(FWHM max (um))
D Median(FWHM min (um))
D Median(FWHM z (um))
I Count(path_image)
S server path
S output location
D frame size
D pixelsize x
D pixelsize y

Flow Variable List
i currentIteration
i maxIterations
s Location
s URL
s RowID
s knime.workspace

Category: All

Function
capitalize(str)
capitalize(str, chars)
compare(str1, str2)
count(str, toCount)
count(str, toCount, modifiers)
countChars(str, chars)
countChars(str, chars, modifiers)
indexOf(str, toSearch)
indexOf(str, toSearch, modifiers)
indexOf(str, toSearch, start)
indexOf(str, toSearch, start, modifiers)

Description

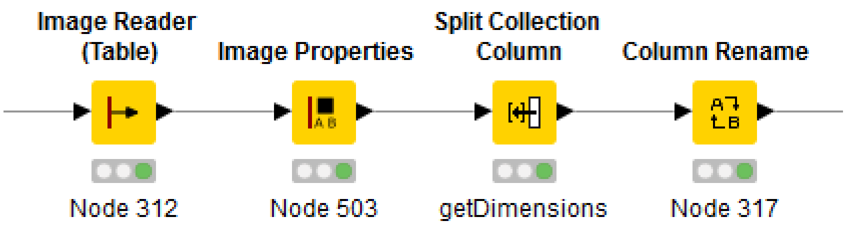
Expression
```
replace($Location$, ".csv", ".tif")
```

Append Column: [✓ Replace Column: D FWHM z (um)]

☐ Insert Missing As Null
☑ Syntax check on close

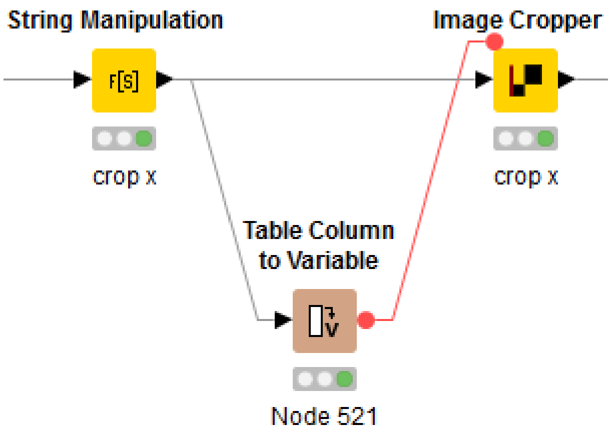
OK | Apply | Cancel | ?

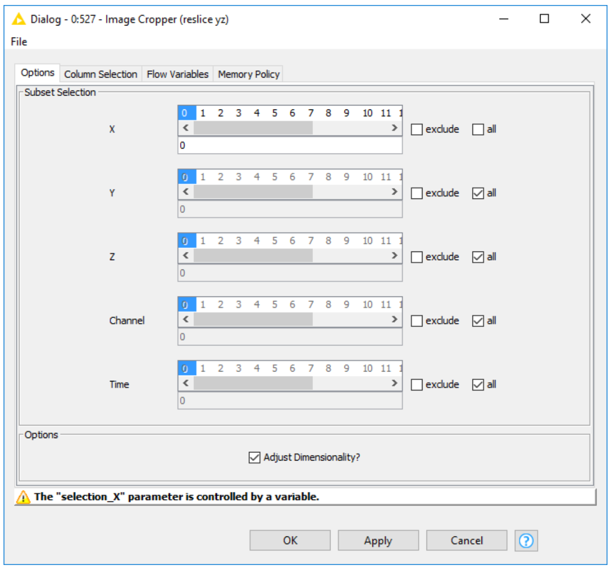
Once the image file is read in, ***Image Properties*** allows extracting information like the dimensions of the image or pixel size. The three dimensions x,y,z are fused into one so-called collection column. The single dimensions are extracted using ***Split Collection Column. Split Collection Column*** splits the collection column into three indivual columns – one for each dimension in x,y, z. ***Split Collection Column*** automatically names the created colums (Split Value 1, Split Value 2, Split Value 3). The columns are then renamed using ***Column Rename***.



Cropping an image using variable coordinates

Each image of the sample bead has to be cropped around the coordinates of its center. For this, the coordinates of the cropped window are passed as string variable to the ***Image Cropper***. The coordinates of each dimension (x,y,z) are treated separately. Example: if a bead with an x coordinate of 30 should be cropped, such that 19 pixels are left on both sides. All pixels with an X-coordinate of x=10 until x=50 should be kept. ***Image Cropper*** needs therefore the information `10-50` as input. In this example. First a string variable would be generated with the content `10-50`. The generated string variable is then passed via ***Table Column to Variable*** to ***Image Cropper*** (Image Cropper > Option > X > untick "all"; Image Cropper > Option > Flow Variables).





Dialog - 0:527 - Image Cropper (reslice yz)

File

Options | Column Selection | Flow Variables | Memory Policy

Subset Selection

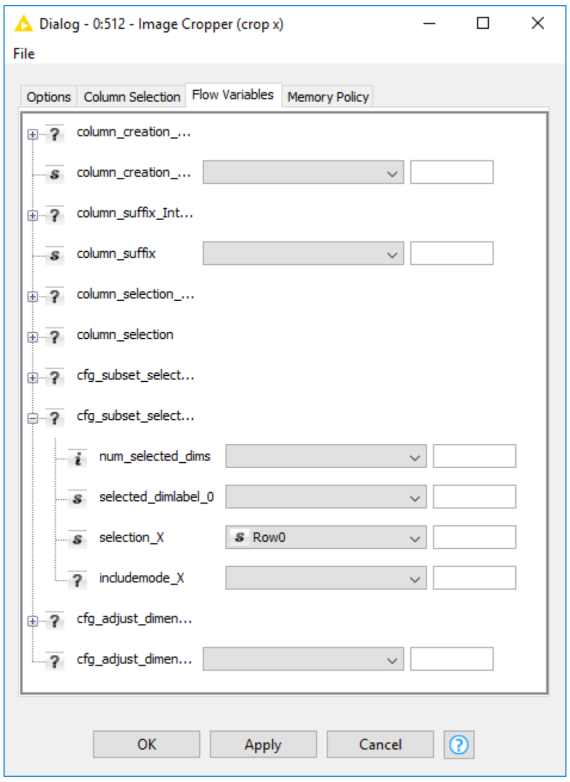
X	0	[ ] exclude	[ ] all
Y	0	[ ] exclude	[✓] all
Z	0	[ ] exclude	[✓] all
Channel	0	[ ] exclude	[✓] all
Time	0	[ ] exclude	[✓] all

Options

[✓] Adjust Dimensionality?

The "selection_X" parameter is controlled by a variable.

OK | Apply | Cancel



Dialog - 0:512 - Image Cropper (crop x)

File

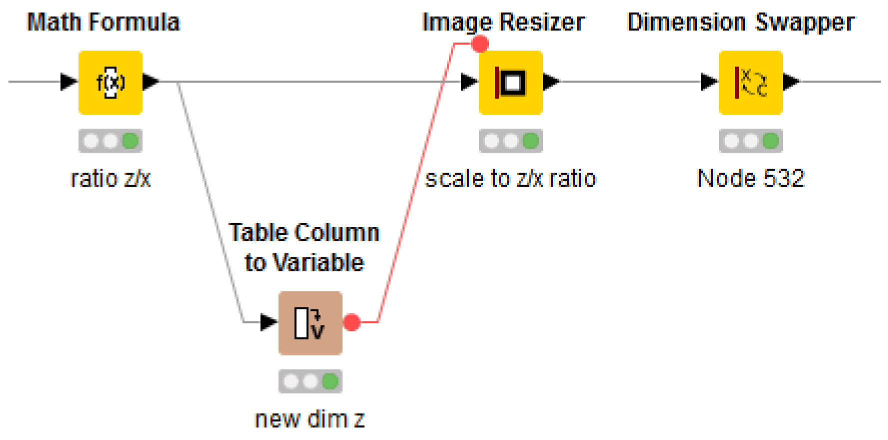
Options | Column Selection | Flow Variables | Memory Policy

```
column_creation_...
column_creation_...
column_suffix_Int...
column_suffix
column_selection_...
column_selection
cfg_subset_select...
cfg_subset_select...
    num_selected_dims
    selected_dimlabel_0
    selection_X          Row0
    includemode_X
cfg_adjust_dimen...
cfg_adjust_dimen...
```

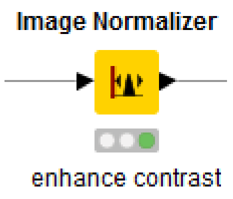
OK | Apply | Cancel

Reslicing images in KNIME

The xz-, and yz-views are generated by ***Image Cropper,*** controlled by the variable as explained above. Next, ***Image Resizer*** compensates for the different pixelsize in x/y and z. To reduce the space needed in the final table, the xz and yz-views are rotated by 90° using ***Dimension Swapper***.



After Cropping in all dimensions and resizing the images, the contrast of the final images are enhanced by ***Image Normalizer*** optimizing the contrast for each image individually.

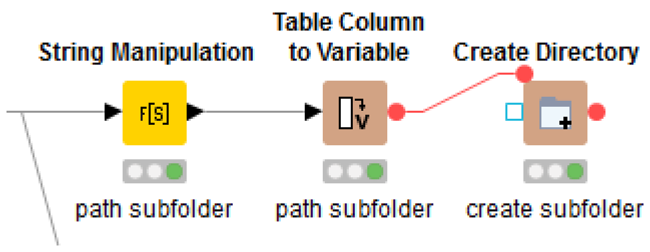


Exporting Data from PSFtracker

PSFtracker saves the following files:

- Images of one example bead for each original image file (cropped xy, xz-, yz-views)
- Plots graphically displaying the FWHM over time
- An HTML-File for publishing on a website which embeds the above images and plots
- XLS-files listing the collected measurements

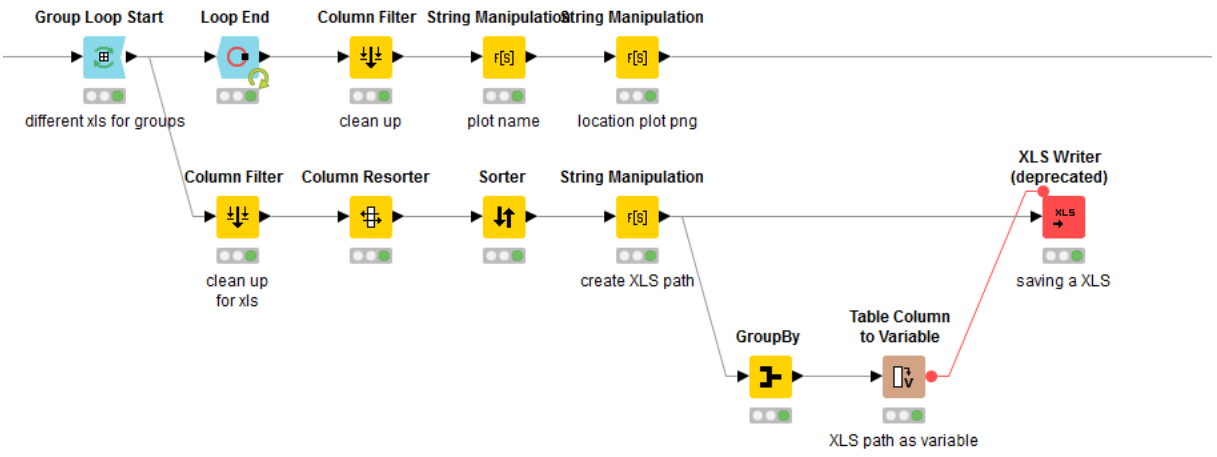
The user specifies an output folder when starting PSFtracker (see above). KNIME generates subfolders for saving the data using ***Create Directory***, using the path to the new directory as variable input.



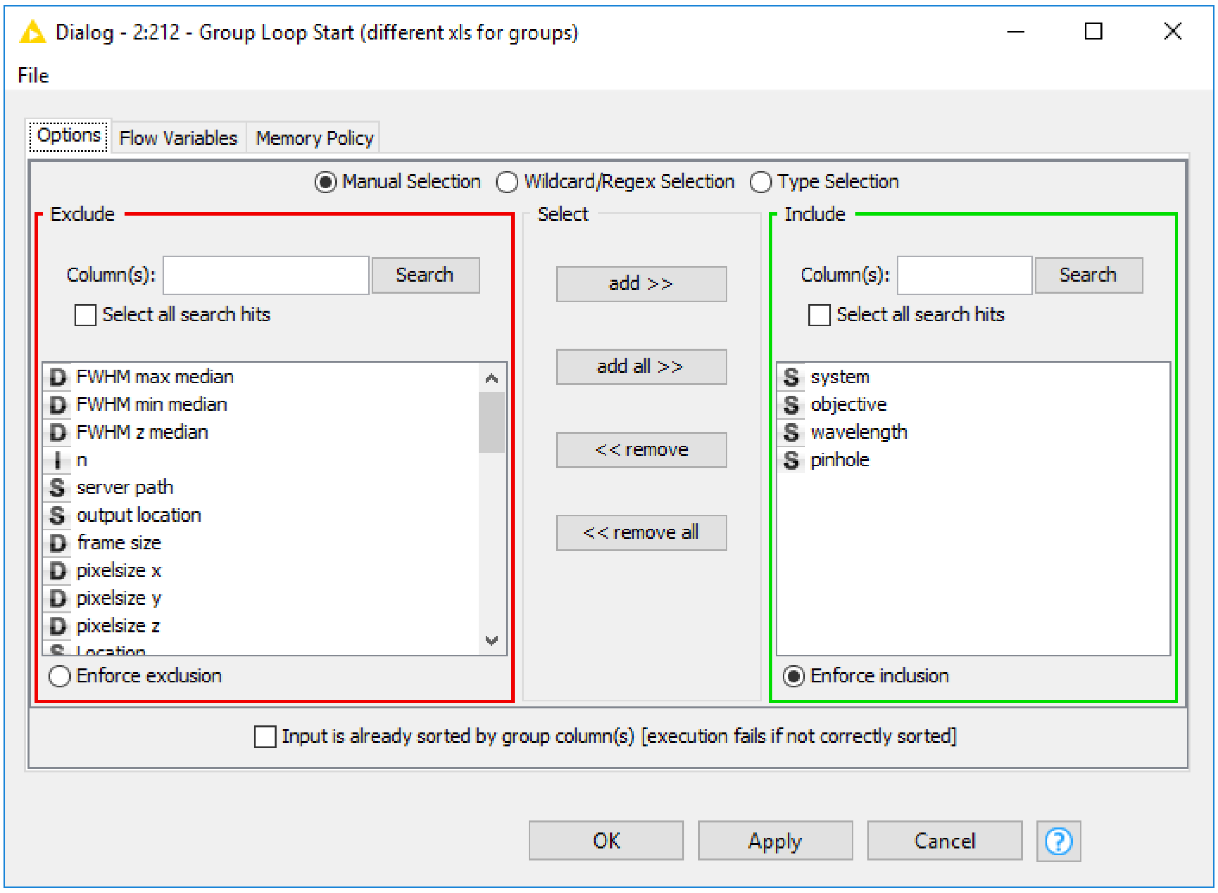
Group handling for saving as XLS and HTML

For some of the KNIME-output, it makes sense to sort the data into different groups. For example, different XLS files should be saved for different microscopes. Sorting is done using

Group Loop Start. ***Group Loop Starts*** constructs groups of data, e.g. measurements from the same system, and all nodes until ***Loop End*** execute one group after another.



Configuration of Group Loop Start to generate groups of data recorded on the same system, using the same objective, wavelength, and pinhole-setting:


Dialog - 2:212 - Group Loop Start (different xls for groups)

File

Options | Flow Variables | Memory Policy

[✓ Manual Selection] Wildcard/Regex Selection Type Selection

Exclude

Column(s): Search

Select all search hits

D FWHM max median
D FWHM min median
D FWHM z median
I n
S server path
S output location
D frame size
D pixelsize x
D pixelsize y
D pixelsize z
S Location

Enforce exclusion

Select

add >>
add all >>
<< remove
<< remove all

Include

Column(s): Search

Select all search hits

S system
S objective
S wavelength
S pinhole

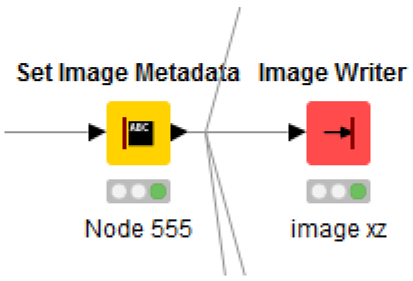
[✓ Enforce inclusion]

Input is already sorted by group column(s) [execution fails if not correctly sorted]

OK Apply Cancel

Saving the images of example beads

The example images are saved by ***Image Writer***. In order to save the xz-, and yz-views, the metadata of these images are first transformed into an xy-2D image. The labels of dimensions 0 and 1 have to be changed to X and Y, respectively.



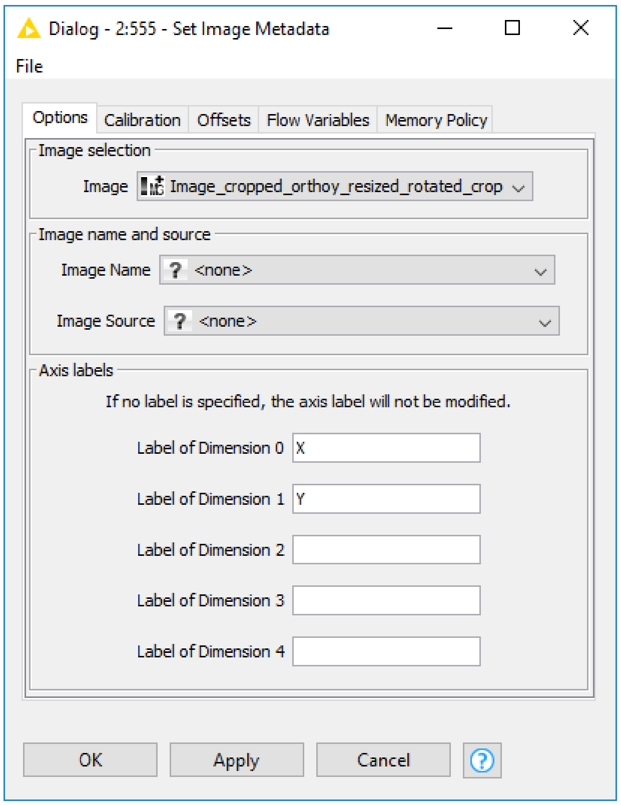


Image selection

Image: Image_cropped_orthoy_resized_rotated_crop

Image name and source

Image Name: <none>

Image Source: <none>

Axis labels

If no label is specified, the axis label will not be modified.

Label of Dimension 0	X
Label of Dimension 1	Y
Label of Dimension 2	
Label of Dimension 3	
Label of Dimension 4	

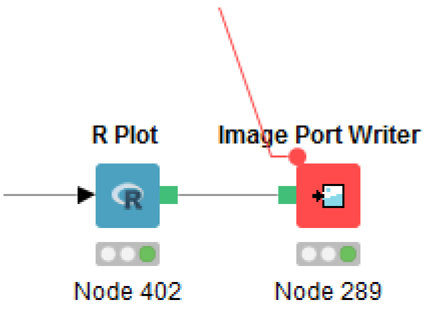
Saving the XLS-File

The path for saving the Excel-sheet – including the filename and file extension - is created using ***String Manipulation***, and passed on via a variable to the XLS Writer. Before saving the table, the data are sorted by ***Column Resorter*** and ***Sorter,*** and also ***Column filter.*** As discussed above, saving of the XLS file is packed within a Group Loop, to ensure individual XLS files for different groups. One group consists of measurements made on the same microscope, using the same objective, wavelength and pinhole-settings.

Generating plots and saving them

To generate plots of the FWHMs over time, KNIME accesses the external program R which has to run in the background. This requires connection to an R-server or a local installation of R. For simplicity this manual assumes a local installation. If a local R-installation is used, open R and run the command `library(Rserve); Rserve(args = "--vanilla").` The access is handled by ***Community Nodes /R Scripting* > *R Plot*** using “Lineplot with plot options” as a template

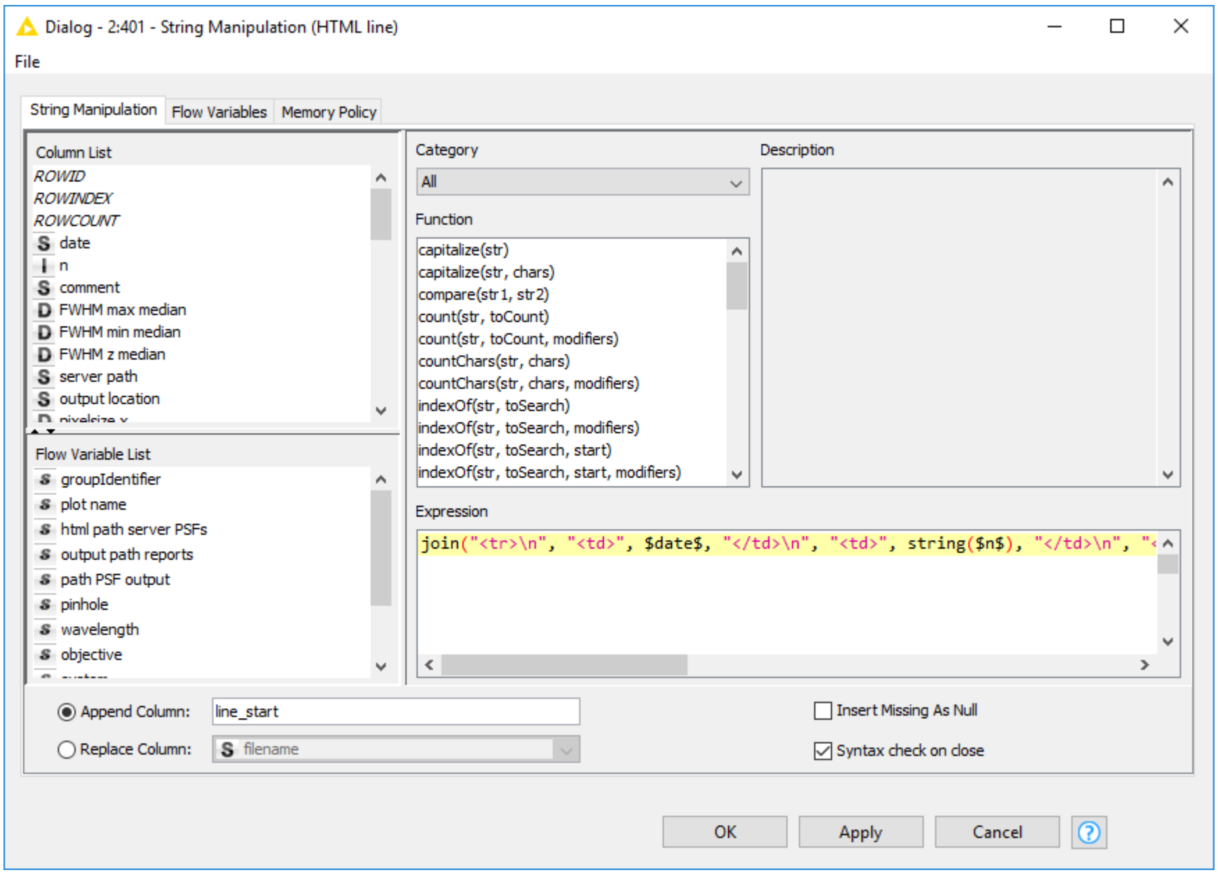
The generated plot is saved using the ***Image Port Writer.*** The path for saving is delivered via a variable. This procedure does not allow to scale the x-axis to real time lengths that have passed between measurements.



Plots display the median FWHM min, max and z for each day an image was recorded. If more than one image was recorded on a given day, results are plotted as average +-StD of all replicates performed on that date (see figure in the main article).

Generating and saving an HTML-file

The KNIME-workflow allows easy publication of the data on a website by generating an HTML-formatted summary file. HTML-code in general contains different tags in addition to the content that allow correct formatting of the text later on the website. For example, the final HTML-file lists the median FWHM-values of different measurement days in a table. Surrounding the FWHM-value by the tags <td> and </td> encodes that the value is put inside a table column. Combining the information with the correct tags is done in KNIME using ***String Manipulation***.


Dialog - 2:401 - String Manipulation (HTML line)

File

String Manipulation | Flow Variables | Memory Policy

Column List
ROWID
ROWINDEX
ROWCOUNT
S date
I n
S comment
D FWHM max median
D FWHM min median
D FWHM z median
S server path
S output location

Flow Variable List
S groupIdentifier
S plot name
S html path server PSFs
S output path reports
S path PSF output
S pinhole
S wavelength
S objective

Category: All

Function
capitalize(str)
capitalize(str, chars)
compare(str1, str2)
count(str, toCount)
count(str, toCount, modifiers)
countChars(str, chars)
countChars(str, chars, modifiers)
indexOf(str, toSearch)
indexOf(str, toSearch, modifiers)
indexOf(str, toSearch, start)
indexOf(str, toSearch, start, modifiers)

Description

Expression
```
join("<tr>\n", "<td>", $date$, "</td>\n", "<td>", string($n$), "</td>\n", "
```

(● Append Column: line_start
○ Replace Column: S filename

☐ Insert Missing As Null
☑ Syntax check on close

OK Apply Cancel

The final html-file is saved using ***CSV writer***, passing the path to the final CSV-file as variable.

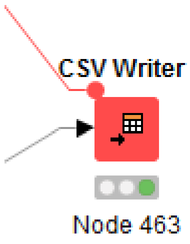


Figure 3 (main text) shows a part of the data after uploading the HTML-File into our local content management system.

Output structure

PSFtracker will automatically create sub-folders in your output path and will save the following files to them:

output folder/system/psf-measurements/bead images/objective

- xy-, xz, and yz-images of one example bead per measurement (see above).

output folder/system/psf-measurements/reports/objective

- **Excel-Tables**: KNIME combines all measurement obtained on the same system using the same objective, wavelength and pinhole-settings. It then saves these collected measurements into one Excel-File with the name system_objective_wavelength_pinhole.xls
- **FWHM plots**: KNIME combines the measurements for FWHM max, FWHM min, and FWHM z obtained on the same system using the same objective, wavelength and pinhole-settings. It saves these data a line plot with the name system_objective_wavelength_pinhole.png.
- **HTML-File**: KNIME combines all measurement obtained on the same system and using the same objective into one final HTML-File with the name system_objective.html. The data can thus be easily uploaded onto your web site.

Reference:

Schindelin, J., I. Arganda-Carreras, E. Frise, V. Kaynig, M. Longair, T. Pietzsch, S. Preibisch, C. Rueden, S. Saalfeld, B. Schmid, J.Y. Tinevez, D.J. White, V. Hartenstein, K. Eliceiri, P. Tomancak, and A. Cardona. 2012. Fiji: an open-source platform for biological-image analysis. *Nat Methods*. 9:676-682.

Change log:

2020-02-25: minor changes in wording, to clarify some things. Added KNIME version 4.1.1 and R 3.6.2 as apparently working.