Supporting Information for the global RNA-RNA interactome of Klebsiella pneumoniae unveils a small RNA regulator of cell division

Eric Ruhland, Malte Siemers, Ruman Gerst, Felix Späth, Laura Vogt, Marc-Thilo Figge, Kai Papenfort, Kathrin Sophie Fröhlich

corresponding author: Kathrin Sophie Fröhlich
Email: kathrin.froehlich@uni-jena.de

1 Summary

1.1 Methods

1.1.1 Image analysis

The images were provided in a 16-bit CZI format and automatically analyzed via a purpose-built image analysis workflow (see Figures SI1-SI9) designed in the ImageJ-based visual programming language JIPipe (https://www.jipipe.org) [Gerst, Cseresnyes et al. 2023, Nat Methods]. The output comprises a table containing the lengths of each cell, organized by metadata extracted from the image name. The pipeline starts with the detection of the CZI files in the input direction and extraction and organization of metadata into the JIPipe annotation format. The tagged images files are then imported via BioFormats [Linkert et al. 2010, JCB] and further processed by extracting the image channel that contains the transmitted light microscopy data. The resulting image is processed by two sets of operations. The first branch detects the bacteria and includes a background subtraction, noise removal and automated thresholding. As the detection method is prone to clustering cells in close vicinity, the second branch segments the halo around each object and subtracts those from the results of the cell thresholding. To resolve overlaps and measure cell lengths, the segmented objects are transformed into a graph structure that contains information about the cell's medial axis and diameter. This simplified representation allowed the usage of an algorithm to split overlapping clusters into individual objects by the removal of junction vertices and addition of alternative edges minimize the cell curvature. Finally, the length of each graph component and thus bacterial cell is determined by a JIPipe-provided operation. For the detailed explanation of the workflow and its individual components, as well as the utilized software versions, see below.

2 Supplements

2.1 Methods

2.1.1 Image analysis

The image analysis starts with the raw input image that is thresholded two times (see Figure SI1): the first operation finds the cells, while the second operations detect the halo regions around the objects. The latter is subtracted from the cell mask to split clusters. The final segmentation result is converted into a filament structure that is further processed to resolve overlaps, remove filaments touching the image borders, and annotate the data with cell diameters. This allows for the generation of the bacterial cell length measurements. Finally, visualizations and plots are generated.

The image analysis was implemented in JIPipe (https://www.jipipe.org) [Gerst, Cseresnyes et al. 2023, Nat Methods] version 1.80.0.

The pipeline is organized into the seven functional units “File handling”, “Thresholding”, “Filament analysis”, “Measurements”, “Tables export”, “Plots export”, and “Visualization export” (see Figure SI2). The purpose of these compartments and their underlying functional nodes will be explained in the following paragraphs.

File handling

The purpose of the “File handling” compartment (see Figure SI3) is the detection of CZI image files in the provided directories and the extraction of essential metadata from the file name. Afterwards, the images are imported into ImageJ and the channel containing the transmitted light microscopy data is extracted.

While the required nodes and settings differ greatly due to the changes in file naming schemata, all operations follow a basic schema:

1. A “Folder list” node that contains the path to the input directory. The purpose of this node is to convert the path into a data item that can be processed with JIPipe.
2. The “Add path to annotations” node annotates the path with its name, thus ensuring that the information about the experiment name is not lost in subsequent steps.
3. The input path is processed with a “List files” node, configured to detect CZI files recursively within the input folder. The output is a list of all detected CZI files. According to the default behavior of JIPipe, the annotation from step 2 is also present at each image file.
4. A second “Add path to annotations” node attaches the file name to the image file path.
5. The file name is processed in a “Set/Edit annotations” node that is configured to extract the name, treatment, replicate number, and image number of each CZI path. The exact settings of this operation differ from experiment to experiment due to changes in the file naming scheme.
6. The annotated image file is imported as image data via the “Import image” node.
7. A “Split channels” node, configured to extract the first channel ensures that only the transmitted light microscopy data is processed.
8. Finally, “Remove LUT” is utilized to remove false coloring from the image.

9. The result of “Remove LUT” is the result of the “File handling” compartment.

Thresholding

The “Thresholding” compartment (see Figure SI4) processes the image produced by “File handling” to segment the cells from the background. To avoid accidental clustering due to proximity of certain cells, the processing is split into two branches: the first branch segments the cells, but with the possibility of over detections (i.e., true and false positive results are detected). The second branch finds the halo that surrounds the bacteria (i.e., true negative pixels are targeted). The cell segmentation thus can be improved by subtracting the true negative pixels from the results of the first branch, which are obtained as following:

1. A “Convert image to 8-bit” operation is applied to the result of the “File Handling” compartment, as ImageJ auto-thresholding operations operate only on 8-bit images.
2. The image is inverted using “Invert colors”, to ensure that cells are bright, while the background is dark.
3. The uneven background is suppressed by applying the “Find or subtract background 2D” node (Rolling ball, dark background, radius = 10, corner correction).
4. A “Median blur 2D” node (radius = 5) reduces noise, while preserving hard edges.
5. Thresholding is applied via the “Auto threshold 2D” operation (“Minimum” method).

The second branch consists of one operation:

1. “Auto threshold 2D” (“Huang” method) is applied to the result of the “File Handling” compartment.

The branches are then merged via an “Image calculator 2D” operation (Function “Subtract”, Input 1 = Left, Input 2 = Right) where the left input is connected to the result of the first branch and the right input is obtained from the result of the second branch. The resulting image is the result of the “Thresholding” compartment.

Filament analysis

The results of “Thresholding” are further processed in the “Filament analysis” compartment (see Figure SI5) that converts the cell masks into a graph-based filamentous representation. This reduction in data allows to split clusters and measure individual cell lengths.

The compartment applies the following operations:

1. The mask generated by the “Thresholding” compartment is converted into a skeleton using “Morphological skeletonize 2D”.
2. Afterwards, the skeleton is converted into a JIPipe filament graph using “Binary skeleton to 2D filaments”. The setting “Zero Z voxel size (2D)” of this node is enabled to ensure that measurements are constrained to two dimensions.
3. The filaments generated by “Binary skeleton to 2D filaments” require smoothing to suppress noise introduced by “Morphological skeletonize 2D”. To suppress these effects, the “Smooth filaments” operation is applied (Factor (X) = 5, Factor (Y) = 5,

Factor (Z) = 5, Location merging function = “AVG(values)”, Prevent cross-object edges).

4. To split clusters, the “Fix overlapping filaments (non-branching)” is introduced. The “Mask” input is connected to the output of the “Thresholding” compartment to prevent the generation of edges outside the cell area. The node automatically removes junctions (vertices with grade > 2) and attempts to bridge connections (candidates) between the newly created end points. To avoid long connections and sharp cell curvatures, the node’s candidate edge filter function is set to ignore candidates with a length greater or equal to 25 px and an angle between the two endpoint directions of less than equal to 130°. The remaining candidates scored by the function “default - MAX(0, length - 10) / 25” to prefer shorter connections.
5. To annotate filament vertices with the thickness of the cell, the output of the “Thresholding” compartment is first pre-processed with “Euclidean distance transform 2D”. The resulting distance map is used as second input for “Set filament vertex radius from image”.
6. Finally, all filaments that cross at least 20 pixels from the image border are removed using the “Remove border filaments” node. The “Reference” input is obtained from the output of the “Thresholding” compartment.

Measurements

The “Measurements” compartment (see Figure SI6) further processes the results of the “Filament analysis” compartment to generate measurements. The following operations are applied:

1. The “Measure filaments” node measures various properties per connected component in the filament graph. As it was ensured during cluster splitting that only vertices are only connected if they are related to the same object, the resulting measurements are created per cell.
2. A “Remove table column” operation removes all measurements except “Component” (the cell ID), and “lengthUnitRadiusCorrected” (the corrected length in µm).
3. The “Component” table column is renamed to “Cell” via the “Rename table column” operation. The resulting table is the final output of the “Measurements” compartment.

Tables export

The “Tables export” compartment (see Figure SI7) saves Excel tables containing the results into a directory “results/<Experiment>_<Date>” or a sub-directory. The pipeline contains three branches that generate different variations of the same outputs.
All branches share a common preprocessing operation “Add annotations as columns” that adds the metadata managed by JIPipe into the output table generated by the “Measurements” compartment.

The first branch exports for each name, treatment, and replicate one Excel table:

1. The image number and file name annotations are removed via “Remove annotation” with the filter “key IN ARRAY("#ImageId", "#Dataset")”. JIPipe now cannot distinguish tables with different image numbers and file names anymore.
2. A “Merge table rows” operation merges the rows of tables that are indistinguishable.

3. The table is exported into an Excel file using “Export table as XLSX”.

The second branch creates one Excel table for all results:

1. A “Merge table rows” operation merges the rows of tables. Its “Merging data batch generation/Grouping method” setting is changed to “All into one batch”, thus generating exactly one table containing all results.
2. The table is exported into an Excel file using “Export table as XLSX”.

The third branch creates one Excel table per image file:

1. The table is exported into an Excel file using “Export table as XLSX”. JIPipe by default will distinguish between tables, thus create one file per image file.

Plots export

The “Plots export” compartment (see Figure SI8) generates five plots that will be explained in the following paragraphs. All results are written into a directory “results/<Experiment>_<Date>” or a sub-directory.

The first plot contains the cell length distribution for each image:

1. The output table generated by the “Measurements” compartment is processed by “Add annotations as columns”, thus adding the metadata managed by JIPipe into the table.
2. The table is processed by the “Histogram plot” node, configured with the series name “#Dataset” (image file name). The values are sourced from the “Length” column.
3. The plot is exported using “Export image”.

The second plot contains the length distribution of all cells in all provided images:

1. The output table generated by the “Measurements” compartment is processed by “Add annotations as columns”, thus adding the metadata managed by JIPipe into the table.
2. A “Merge table rows” operation merges the rows of tables. Its “Merging data batch generation/Grouping method” setting is changed to “All into one batch”, thus generating exactly one table containing all results.
3. The table is processed by the “Histogram plot” node, configured with the series name “#Dataset” (image file name). The values are sourced from the “Length” column.
4. The plot is exported using “Export image”.

The third plot generates a cell length distribution for each treatment:

1. The output table generated by the “Measurements” compartment is processed by “Add annotations as columns”, thus adding the metadata managed by JIPipe into the table.

2. A “Merge table rows” operation merges the rows of tables. It is configured with a custom “Merging data batch generation/Grouping method” and is configured to differentiate tables only by their treatment metadata. This results in as many tables as there are treatments.
3. The table is processed by the “Histogram plot” node, configured with the series name “#Dataset” (image file name). The values are sourced from the “Length” column.
4. The plot is exported using “Export image”.

The fourth plot is a box plot that compares the different treatments:

1. The output table generated by the “Measurements” compartment is processed by “Add annotations as columns”, thus adding the metadata managed by JIPipe into the table.
2. A “Merge table rows” operation merges the rows of tables. It is configured with a custom “Merging data batch generation/Grouping method” and is configured to differentiate tables only by their treatment metadata. This results in as many tables as there are treatments.
3. The “Box plot” operation converts the table into a plot. The node is configured to merge all tables into one batch, thus creating exactly one plot. The data is categorized and grouped by a table column containing the treatment. The values are extracted from the “Length” column.
4. The plot is exported using “Export image”.

The fifth plot is a box plot that compares the results based on treatment and name:

1. The output table generated by the “Measurements” compartment is processed by “Add annotations as columns”, thus adding the metadata managed by JIPipe into the table.
2. A “Merge table rows” operation merges the rows of tables. It is configured with a custom “Merging data batch generation/Grouping method” and is configured to differentiate tables only by their treatment metadata. This results in as many tables as there are treatments.
3. The “Box plot” operation converts the table into a plot. The node is configured to merge all tables into one batch, thus creating exactly one plot. The data is categorized by a table column containing the treatment. Values are grouped by a custom expression that ensures that one bar per name is generated. The values are extracted from the “Length” column.
4. The plot is exported using “Export image”.

Visualization export

Visualizations are generated in the “Visualization export” compartment (see Figure SI9). The results are written into a directory “results/<Experiment>_<Date>” or a sub-directory.

The first visualization showcases the detected filaments next to the cell length distribution:

1. The “Set overlay” operation is applied on the output of the “File handling” compartment and the output of the “Filament analysis” compartment. JIPipe implicitly converts filaments to ImageJ ROI if required.

2. The “Render overlay” node draws the filaments onto the input image, yielding an RGB image.
3. The table generated by the “Measurements” compartment is processed with a “Histogram plot” node, configured with the series name “#Dataset” (image file name). The values are sourced from the “Length” column.
4. The results from step 2 and step 3 are combined into one image using “Montage of input images”.
5. The resulting montage is exported using “Export image”.

The second visualization displays the results of the “Thresholding” compartment:

1. The result of the “Thresholding” compartment is processed via “Connected components labeling 2D”, creating an image that assigns a unique greyscale value for each object.
2. The labels are visualized by the “Convert labels to RGB” node.
3. To obtain a hollow label visualization, the “Thresholding” compartment output is processed with “Morphological operation 2D” (Operation “Gradient”, Radius = 1). The resulting mask is used as “Mask” input for a “Set to color (RGB)” node (Only apply to … “Outside mask”) that processes the output of step 2.
4. A two-input “Blend image” operation draws the result of step 3 on top of the output of the “File handling” compartment. The opacity for the top layer is set to 0.9.
5. The resulting image is exported using “Exprot image”.

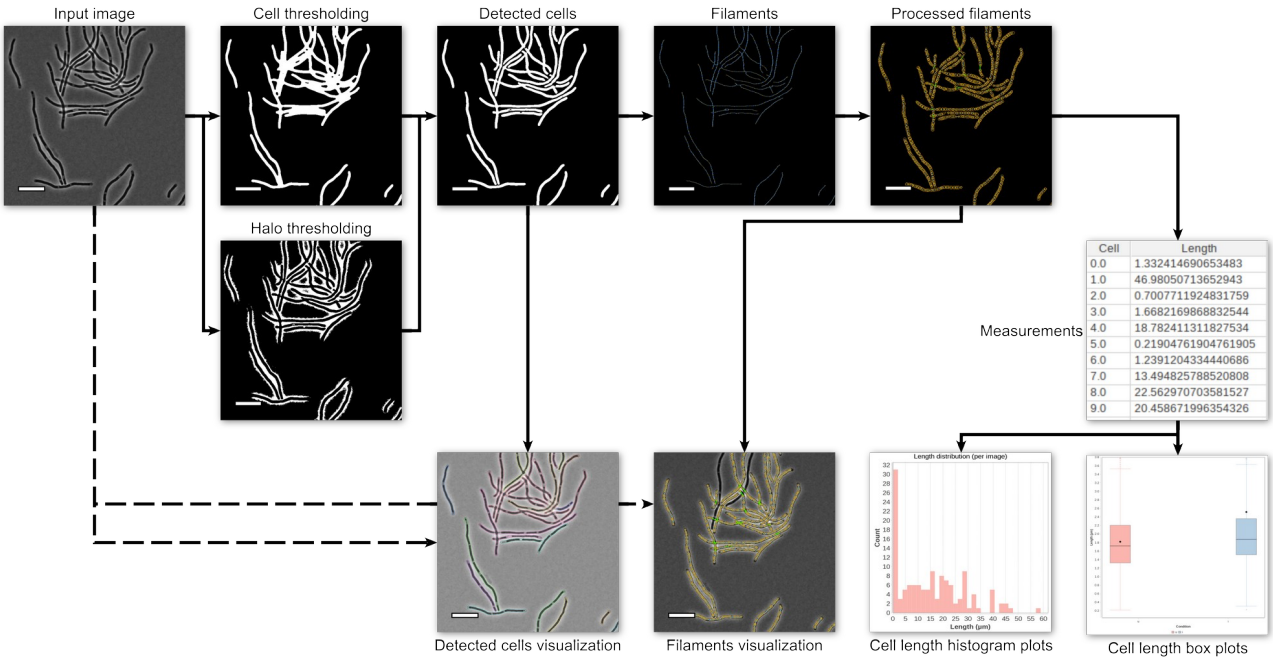

Figure S1 *Overview of the image analysis steps on a representative input. The processing starts with the input image that is segmented in two separate branches, one targeting the cells, and another one segmenting the halos around the cells. The latter are subtracted from the cell mask, which is afterwards converted into filaments. Processed filaments are measured. The detected cells, filaments, and measurements are visualized and plotted. The boxes contain representative images of the processing results. Solid and dashed arrows indicate that the target depends on the output of the source. Scale bars are 8 µm.*

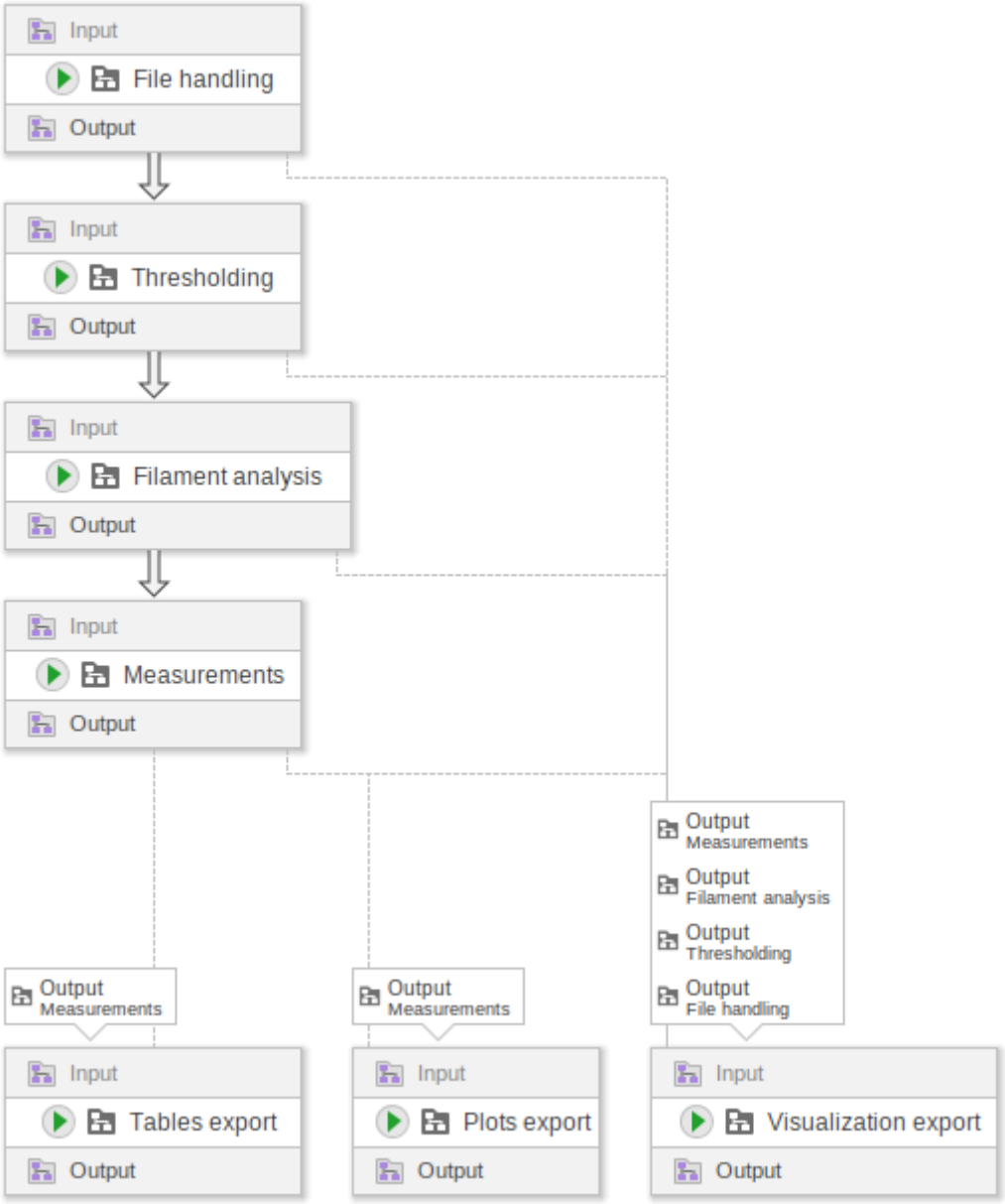

Figure S2 *Overview of the analysis pipeline. White and gray boxes represent functionals unit within the pipeline that have an input (gray area at the top of each node) and an output (gray area at the bottom of each node). Edges (gray arrows) between the boxes indicate that data is transferred between compartments from an output to an input. Certain edges are displayed as dashed line to improve visual clarity. In such cases, the sources of an input are displayed above the input as speech bubble.*

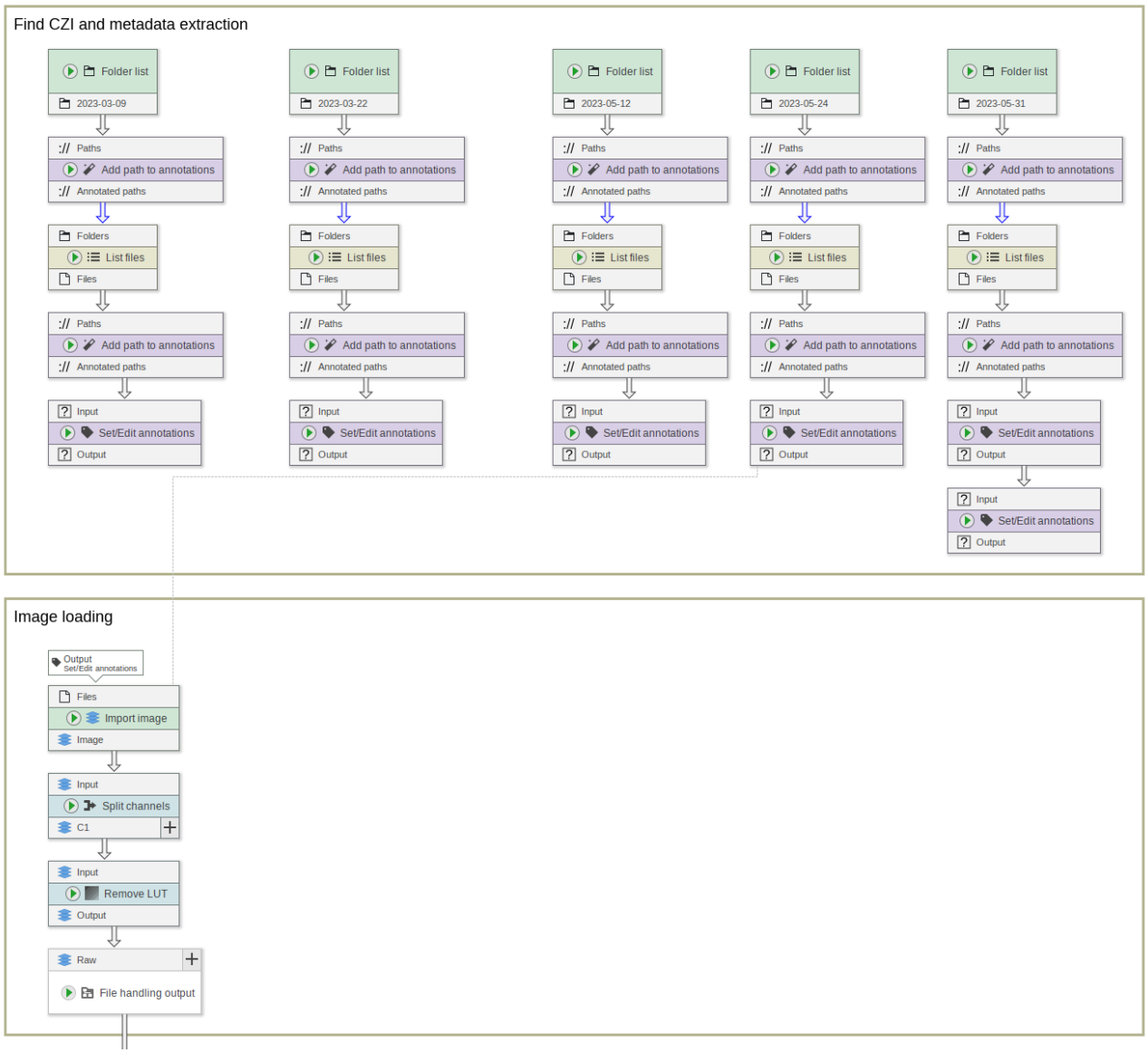

Figure S3 *File handling compartment. The pipeline is divided into two groups: "Find CZI and metadata extraction" contains nodes that are responsible for detecting the CZI image files and extracting metadata. The second group "Image loading" is responsible for importing the files and extracting the channel of interest. The final output is the raw image data. Colored boxes are functional nodes that can contain inputs (gray areas at the top of each node) and outputs (gray areas at the bottom of each node). Edges (gray and blue arrows) between the boxes indicate that data is transferred from an output to an input. Certain edges are displayed as dashed line to improve visual clarity. In such cases, the sources of an input are displayed above the input as speech bubble.*

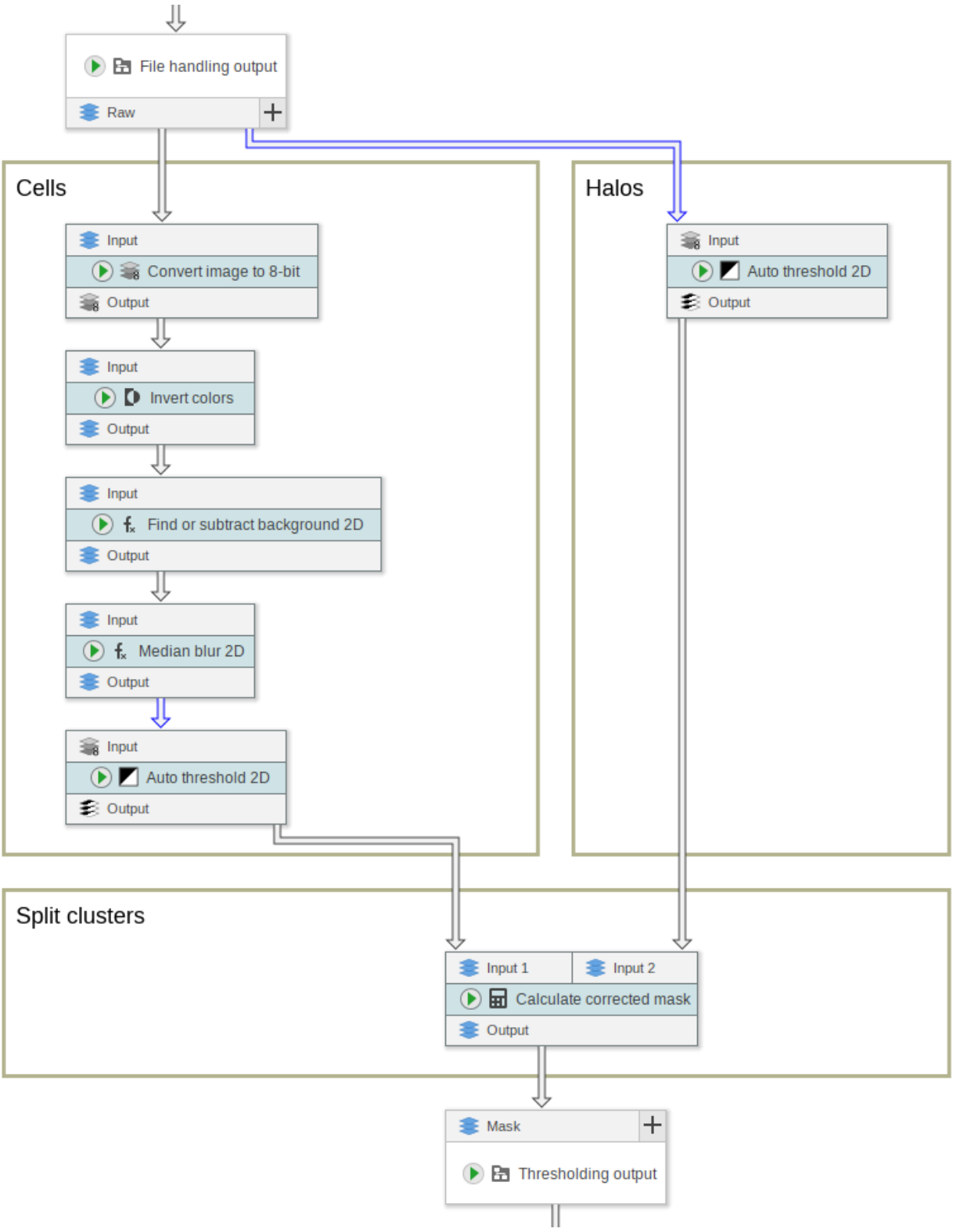

Figure SI4 *Thresholding compartment. Object segmentation is split into two branches: "Cells" contains nodes for detecting the cells, while nodes in "Halos" detect the halos around the cells. The segmented halos are subtracted from the cell segmentations in "Split clusters". Colored boxes are functional nodes that can contain inputs (gray areas at the top of each node) and outputs (gray areas at the bottom of each node). Edges (gray and blue arrows) between the boxes indicate that data is transferred from an output to an input.*

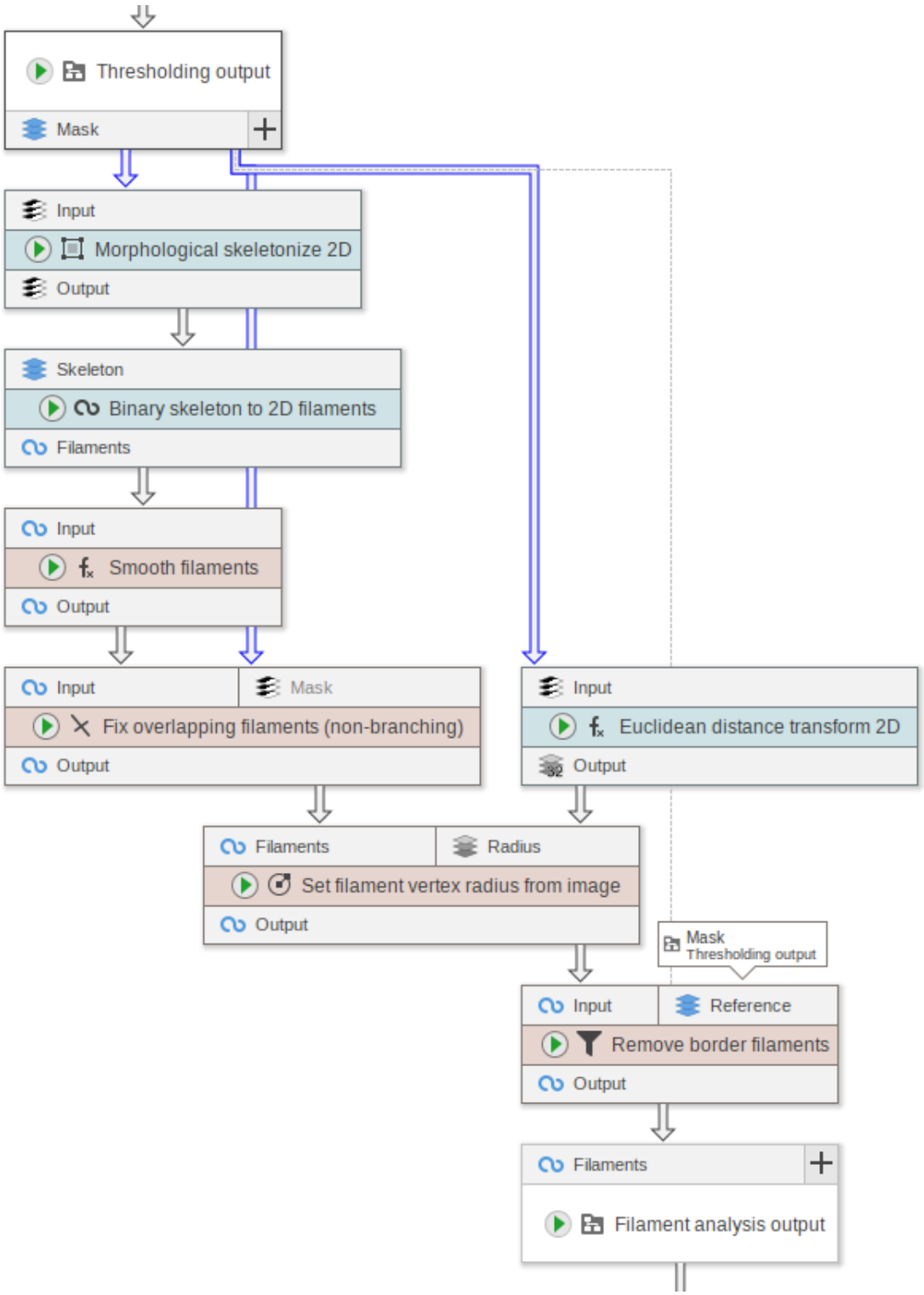

Figure SI5 *Filament analysis compartment. The detected cells are skeletonized and converted into a filament graph. Additional processing includes smoothing, the resolution of overlapping filaments, and assigning vertices with the cell thickness. Colored boxes are functional nodes that can contain inputs (gray areas at the top of each node) and outputs (gray areas at the bottom of each node). Edges (gray and blue arrows) between the boxes indicate that data is transferred from an output to an input. Certain edges are displayed as dashed line to improve visual clarity. In such cases, the sources of an input are displayed above the input as speech bubble.*

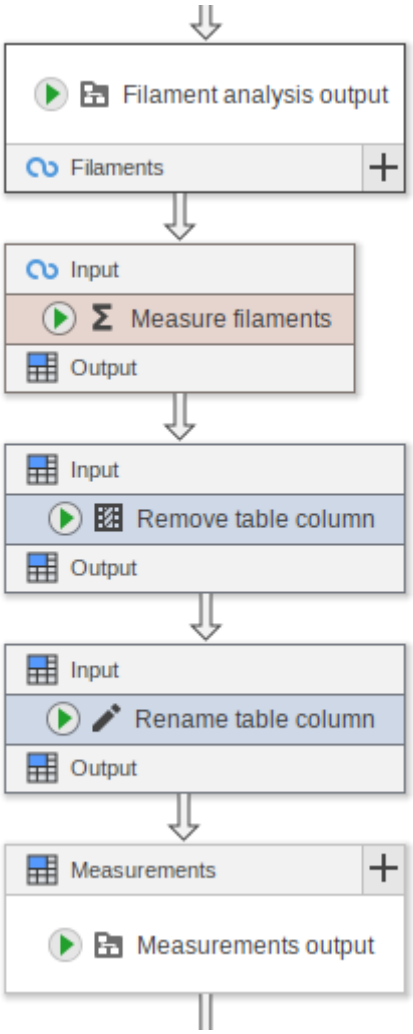

Figure SI6 *Measurement compartment. The filaments are measured, and the resulting table is processed to only contain the identifier of each cell and their lengths. The colored boxes are functional nodes that can contain inputs (gray areas at the top of each node) and outputs (gray areas at the bottom of each node). Edges (gray and blue arrows) between the boxes indicate that data is transferred from an output to an input.*

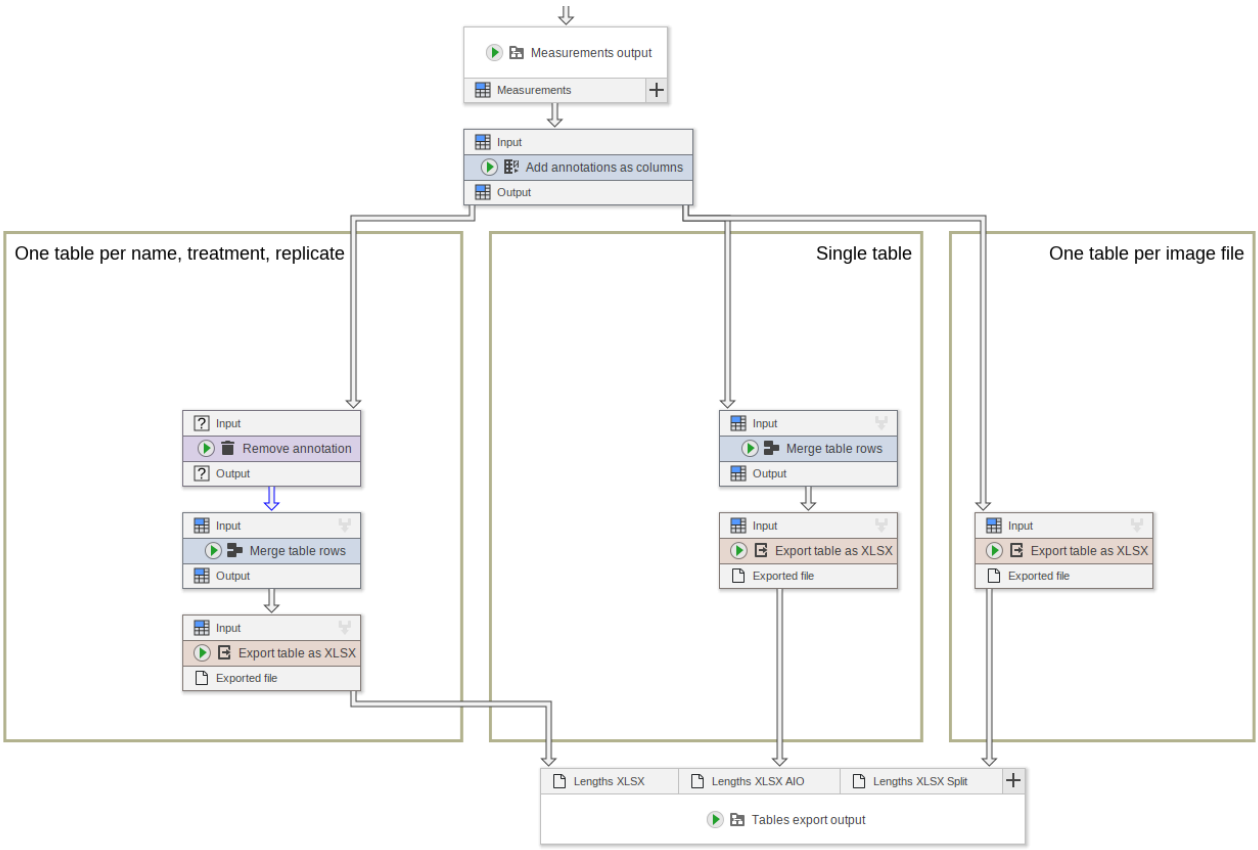

Figure SI7 *Tables export compartment. JIPipe-managed metadata is copied into the cell lengths table in "Add annotations as columns". Afterwards, three outputs are generated. The first result file is created in the "One table per name, treatment, replicate" group and merges the results of multiple image files into one table. A singular output containing all results is written in the "Single table" group. One Excel file per input image is generated in the "One table per image file" group. Colored boxes are functional nodes that can contain inputs (gray areas at the top of each node) and outputs (gray areas at the bottom of each node). Edges (gray and blue arrows) between the boxes indicate that data is transferred from an output to an input.*

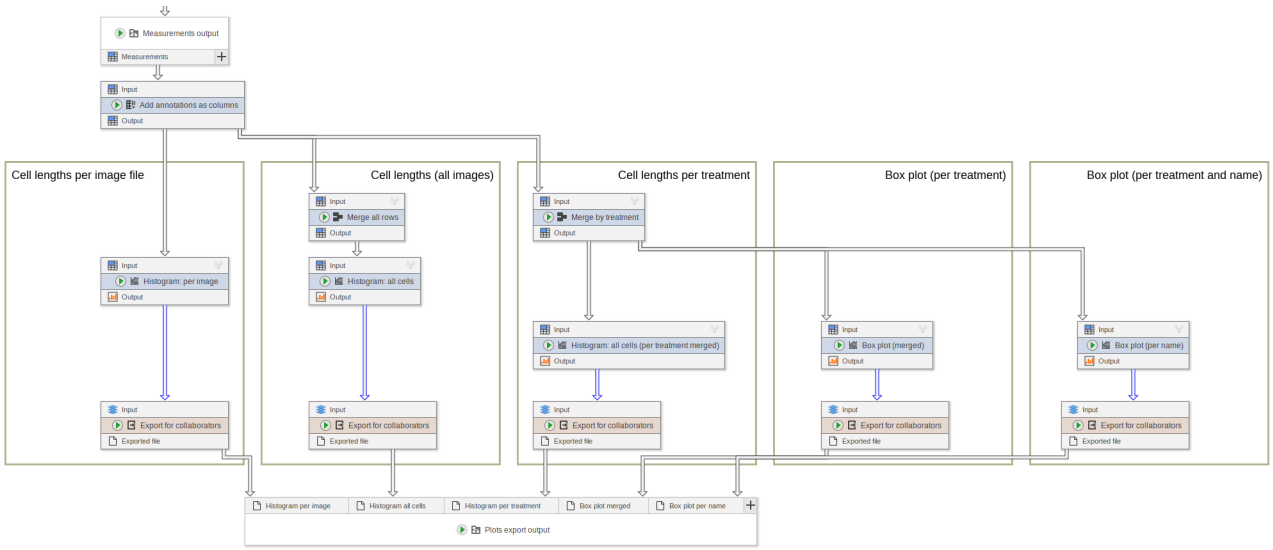

Figure SI8 *Plots export compartment. JIPipe-managed metadata is copied into the cell lengths table in "Add annotations as columns". Five plot types are created by the shown pipeline. Cell length histogram plots are generated per input image file in "Cell lengths per image file", for all images in "Cell lengths (all images)", and for each treatment type in "Cell lengths per treatment". Box plots showcasing the cell lengths per treatment are created in "Box plot (per treatment)", while the nodes in "Box plot (per treatment and name)" build box plots that separate the cell lengths by treatment and name. Colored boxes are functional nodes that can contain inputs (gray areas at the top of each node) and outputs (gray areas at the bottom of each node). Edges (gray and blue arrows) between the boxes indicate that data is transferred from an output to an input.*

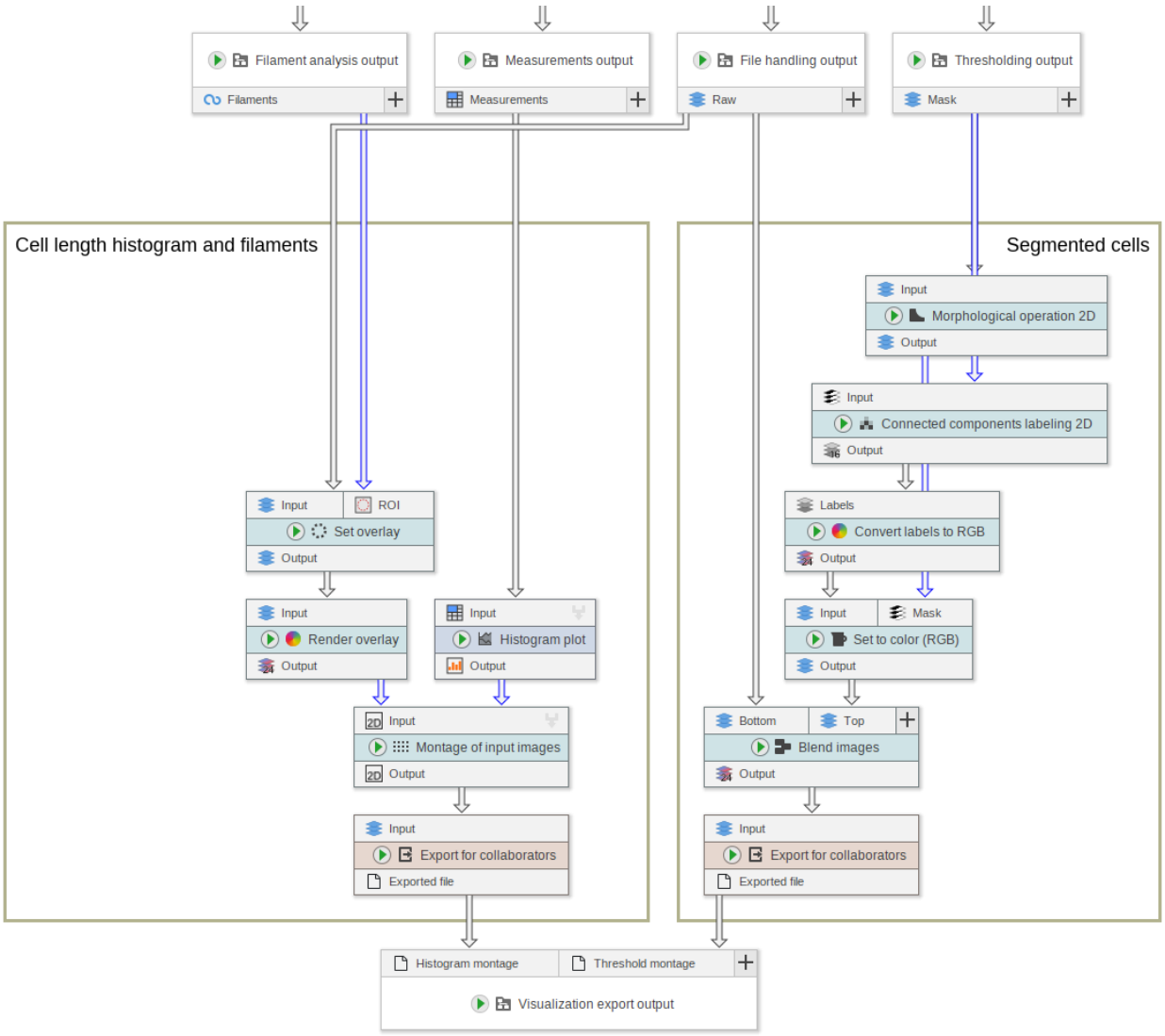

Figure SI9 *Visualization export compartment. Two output types are generated that are colored boxes are functional nodes that can contain inputs (gray areas at the top of each node) and outputs (gray areas at the bottom of each node). Edges (gray and blue arrows) between the boxes indicate that data is transferred from an output to an input.*