nature methods SUPPLEMENTARY INFORMATION
https://doi.org/10.1038/s41592-019-0501-0

In the format provided by the authors and unedited.

BigStitcher: reconstructing high-resolution image datasets of cleared and expanded samples

David Hörl[1,2,7], Fabio Rojas Rusak[3,7], Friedrich Preusser[2], Paul Tillberg[4], Nadine Randel[4], Raghav K. Chhetri[4], Albert Cardona[4,5], Philipp J. Keller[4], Hartmann Harz[1], Heinrich Leonhardt[1], Mathias Treier[3,6] and Stephan Preibisch[2]*

[1]Department of Biology II, Ludwig-Maximilians-Universität München, Munich, Germany. [2]Berlin Institute for Medical Systems Biology, Max Delbrück Center for Molecular Medicine in the Helmholtz Association, Berlin, Germany. [3]Cardiovascular and Metabolic Sciences, Max Delbrück Center for Molecular Medicine in the Helmholtz Association, Berlin, Germany. [4]Janelia Research Campus, Howard Hughes Medical Institute, Ashburn, VA, USA. [5]Department of Physiology, Development and Neuroscience, University of Cambridge, Cambridge, UK. [6]Charité-Universitätsmedizin Berlin, Berlin, Germany. [7]These authors contributed equally: David Hörl, Fabio Rojas Rusak. *e-mail: stephan.preibisch@mdc-berlin.de

BigStitcher: Reconstructing high-resolution image datasets of cleared and expanded samples

Supplementary Figures, Tables, Notes

David Hörl[1,2,#], Fabio Rojas Rusak[3,#], Friedrich Preusser[2], Paul Tillberg[4], Nadine Randel[4], Raghav K. Chhetri[4], Albert Cardona[4,5], Philipp J. Keller[4], Hartmann Harz[1], Heinrich Leonhardt[1], Mathias Treier[3,6], Stephan Preibisch[2,*]

[1]Department of Biology II, Ludwig-Maximilians-Universität München, München, Germany
[2]Berlin Institute for Medical Systems Biology (BIMSB), Max Delbrück Center for Molecular Medicine
in the Helmholtz Association (MDC), Berlin, Germany
[3]Cardiovascular and Metabolic Sciences, Max Delbrück Center for Molecular Medicine
in the Helmholtz Association (MDC), Berlin, Germany
[4]Janelia Research Campus, Howard Hughes Medical Institute, Ashburn, VA, USA
[5]Department of Physiology, Development and Neuroscience, University of Cambridge, Cambridge, UK
[6]Charité-Universitätsmedizin Berlin, Berlin, Germany
[#]D.H. and F.R.R. contributed equally

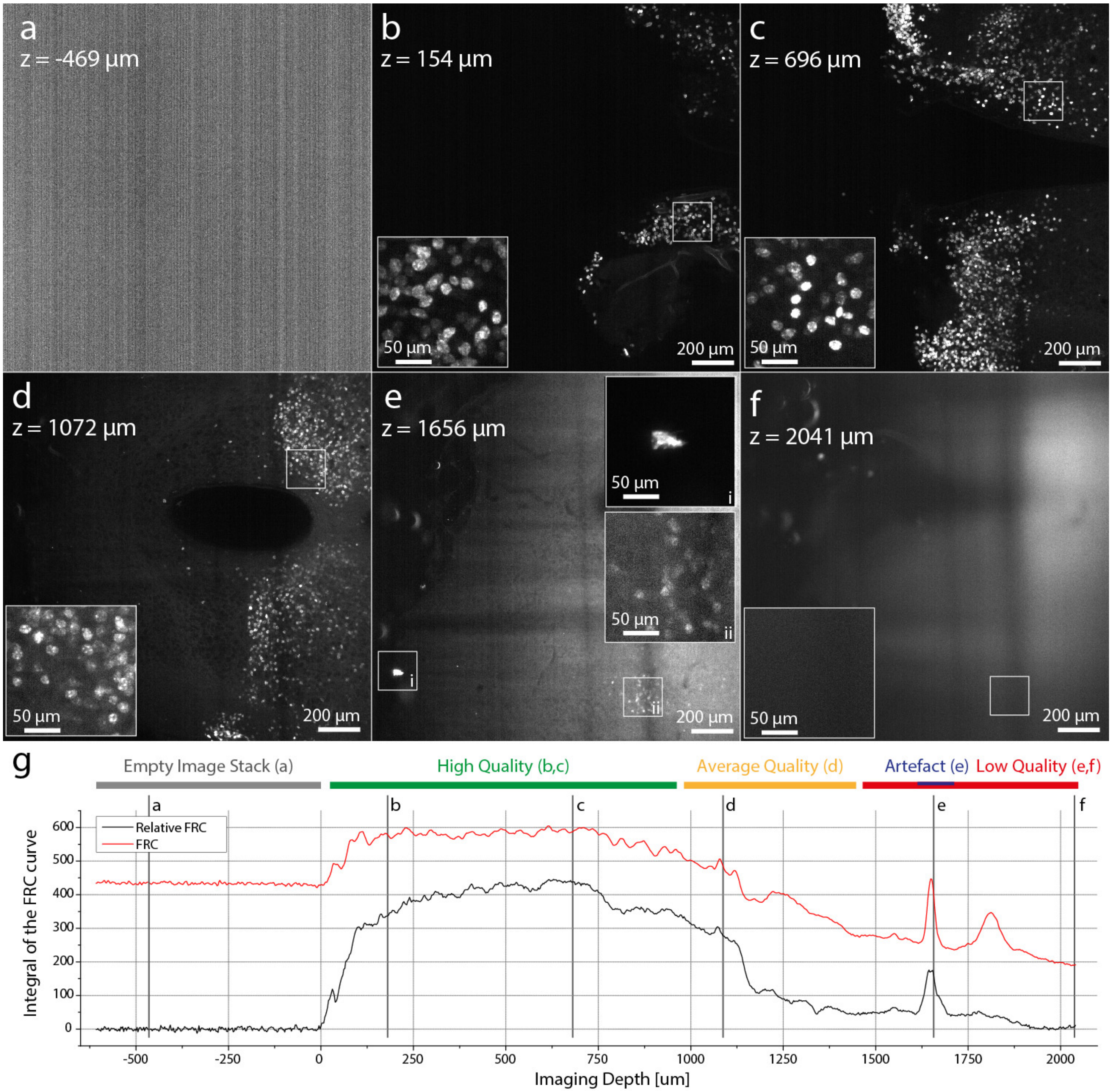
* Correspondence should be addressed to S.P. (stephan.preibisch@mdc-berlin.de)

Continued on the following page

Note: Supplementary Videos 1–9 are available for download on the journal homepage.

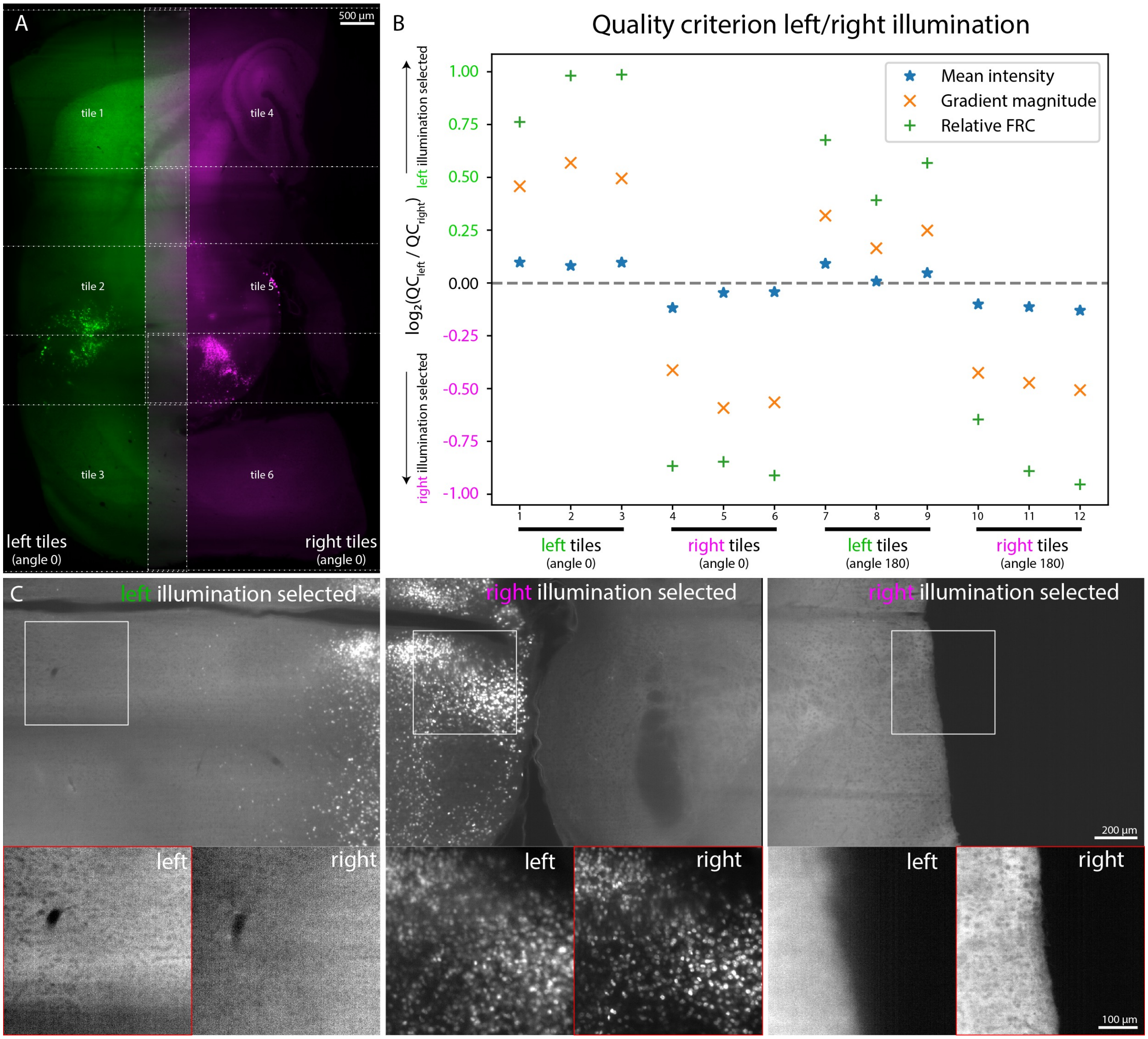
SUPPLEMENTARY FIGURES

SUPPLEMENTARY FIGURE 1: Quantification of fluorescence preservation in cleared tissue



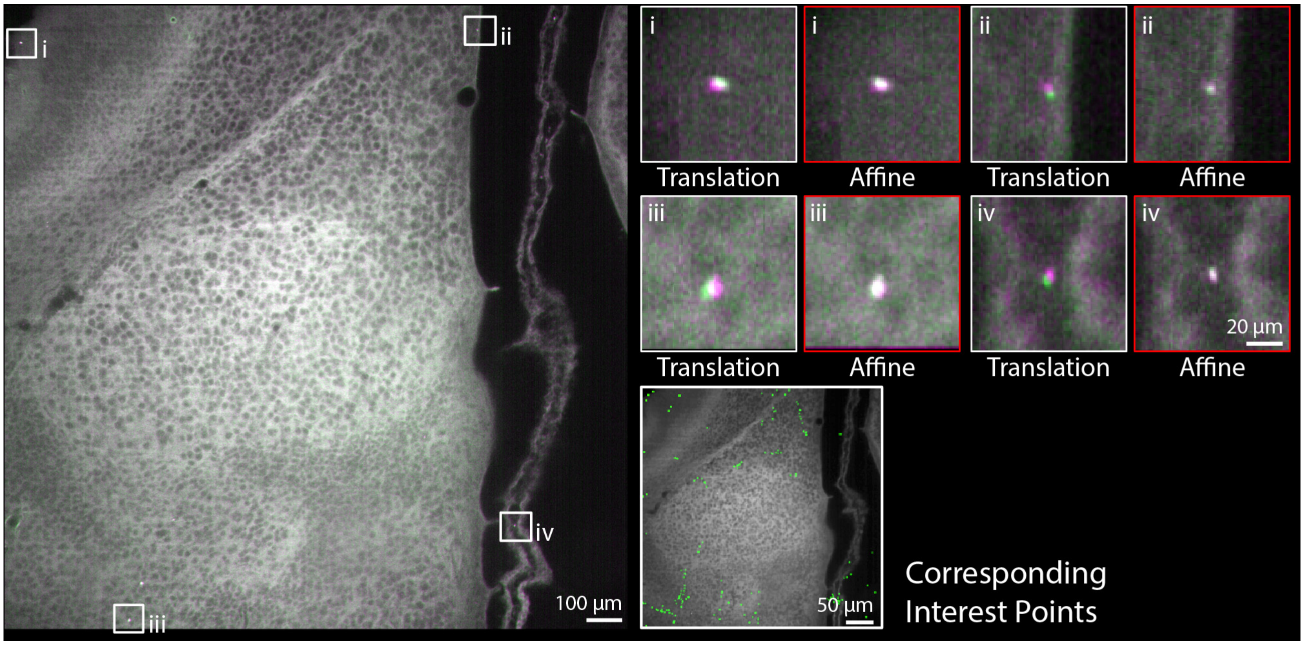
Supplementary Figure 1: *Quantification of fluorescence preservation in cleared tissue.* **(a-f)** Optical sections through an CLARITY-cleared adult mouse hypothalamus expressing H2B-GFP in all bsx neurons. Fluorescence is preserved throughout the clearing procedure. However, the signal is degrading with imaging depth and can typically be recorded up to 1 – 2 cm into the sample, depending on the tissue type and the quality of the clearing process limiting the size of the sample that can be acquired from a single orientation. Brightness and contrast was adjusted individually. **(g)** Quantification of image quality using (relative) Fourier Ring Correlation ([r]FRC, see **Online Methods**) in BigStitcher. Note that FRC produces high values for the camera patterns if no signal is present. The rFRC accurately measures image quality as illustrated by the position of the panels (a-f). As part of this publication similar experiments were performed 4× with comparable clearing results (**Fig. 1n, 3b, 3d**).

SUPPLEMENTARY FIGURE 2: Quantification of automatic illumination selection



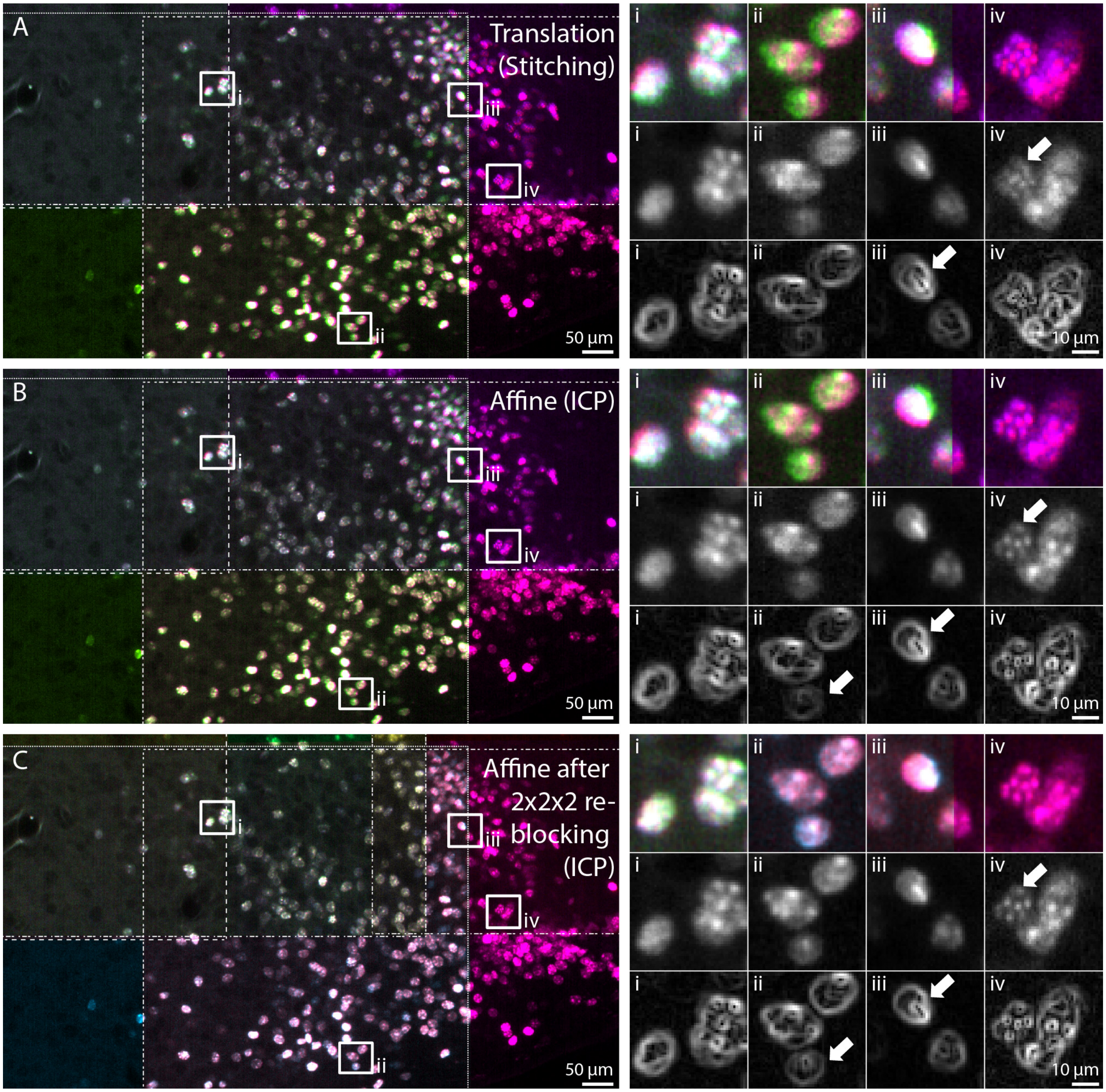
Supplementary Figure 2: *Automatic illumination selection.* **(A)** A small 166GB multi-view, dual-illumination, multi-tile dataset specifically acquired for verification purposes, here to quantify the properties of automatic illumination selection. Shown is a slice through the first of two angles, where six tiles and two illumination directions are highlighted. We manually confirmed that the left three tiles need to be assigned to left illumination, and the right three tiles to right illumination. **(B)** compares the distinction power of the methods *Mean intensity*, *Gradient magnitude*, and *Relative Fourier Ring Correlation* (see **Online Methods**) by their respective quality scores. All methods correctly predict the assignment, while the *Relative Fourier Ring Correlation* distinguishes the illumination directions best. Note that *Mean intensity* almost produces an error for tile 8 of the second angle (180 degrees) **(C)** Another example of best illumination for three consecutive tiles (left to right), selected based on *Mean intensity* for each tile. Close-ups shows the specified region for both illumination directions. **(A,C)** As part of this publication automatic illumination selection was performed on 4 datasets (see also **Fig. 1d, 3b, 3d**).

SUPPLEMENTARY FIGURE 3: Chromatic aberration correction



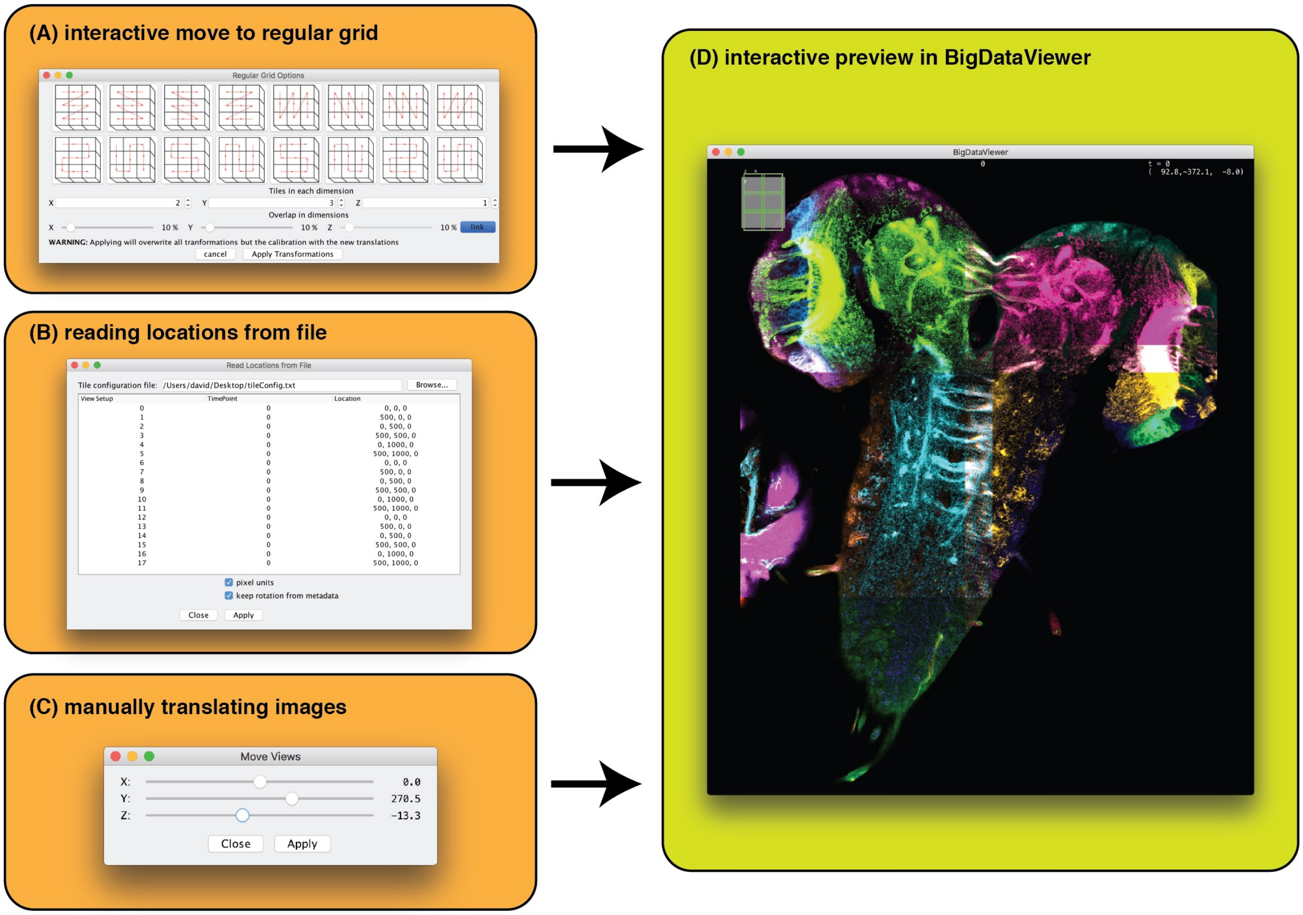
Supplementary Figure 3: *Chromatic aberration correction.* If sufficient autofluorescent signal is in common between channels the ICP refinement on an affine model can be used to approximately correct chromatic aberrations are within the range of a few pixels. Here, the 488 and 561 channels are shown in magenta and green, respectively. Zoom-ins (i) – (iv) illustrate the correction on one example image tile 1920×1920 pixels in size. In the bottom right the interest points (all points of the entire stack are shown for one slice) used for alignment are shown. Please note that for example the point in zoom-in (ii) was not used for alignment. If aberrations are significantly bigger than illustrated in this example or if not enough common autofluorescence between channels exist, images can be preprocessed with dedicated chromatic aberration software before import into BigStitcher (see Limitations section in **Online Methods**). Chromatic aberration correction was applied to all 26 image tiles of this dataset (**Supplementary Video 3**) as well as to all cleared samples that were acquired with 2 channels (**Supplementary Table 1**).

SUPPLEMENTARY FIGURE 4: Spherical aberration correction



Supplementary Figure 4: *Spherical aberration correction.* **(A-C)** show the same area a cleared adult mouse hypothalamus expressing H2B-GFP in all bsx neurons where the corners of 4 image tiles of the same wavelength overlap. Zoom-ins (i)–(iv) show the alignment quality by overlaying different colors (1st row), after image fusing using blending (2nd row), and the sobel-filtering of the blending fusion (3rd row). **(A)** shows results for stitching, **(B)** when using affine refinement, and **(C)** when using affine refinement on re-blocked images. Note that affine, and split-affine improve the alignment quality. Blending-fusion can reduce artifacts as it reduces the contribution of pixels close to image borders. Arrows outline cases where artifacts persist after blending-fusion. E.g., the artifact visible in the fusion in (iii) stems from misalignments of the pink and red tile, since the green tile is almost invisible after fusion. **(A-C)** Spherical aberration correction using affine transformations achieving similar results was applied to all 26 tiles of the dataset, as well as the datasets shown in (**Fig. 1n, 3b-d**).

SUPPLEMENTARY FIGURE 5: Manual alignment



Tiles in each dimension

X 2 Y 3 Z 1

Overlap in dimensions

X 10 % Y 10 % Z 10 %

WARNING: Applying will overwrite all tranformations but the calibration with the new translations

Tile configuration file: /Users/david/Desktop/tileConfig.txt

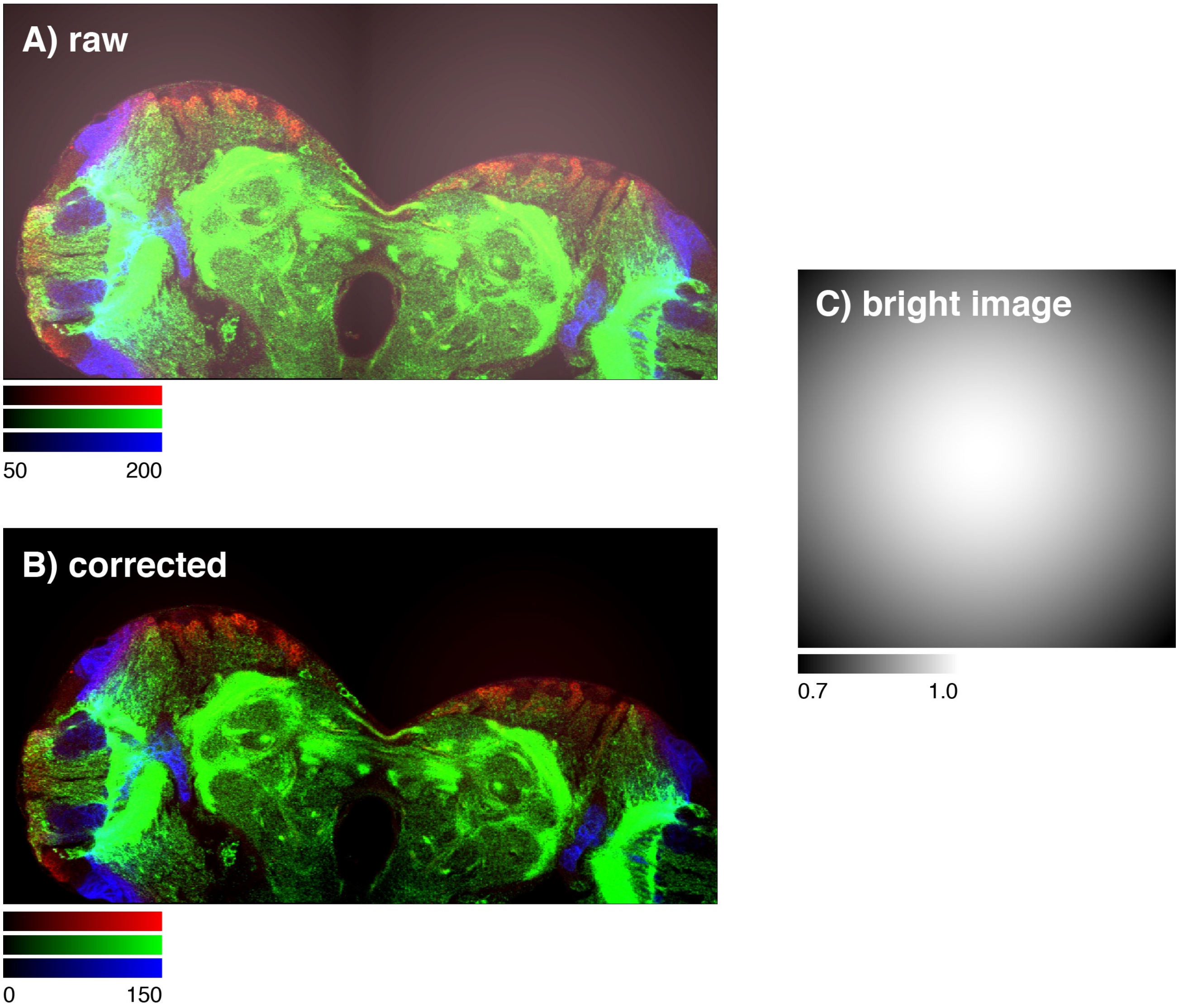
View Setup	TimePoint	Location
0	0	0, 0, 0
1	0	500, 0, 0
2	0	0, 500, 0
3	0	500, 500, 0
4	0	0, 1000, 0
5	0	500, 1000, 0
6	0	0, 0, 0
7	0	500, 0, 0
8	0	0, 500, 0
9	0	500, 500, 0
10	0	0, 1000, 0
11	0	500, 1000, 0
12	0	0, 0, 0
13	0	500, 0, 0
14	0	0, 500, 0
15	0	500, 500, 0
16	0	0, 1000, 0
17	0	500, 1000, 0

☑ pixel units
☑ keep rotation from metadata

X: 0.0
Y: 270.5
Z: -13.3

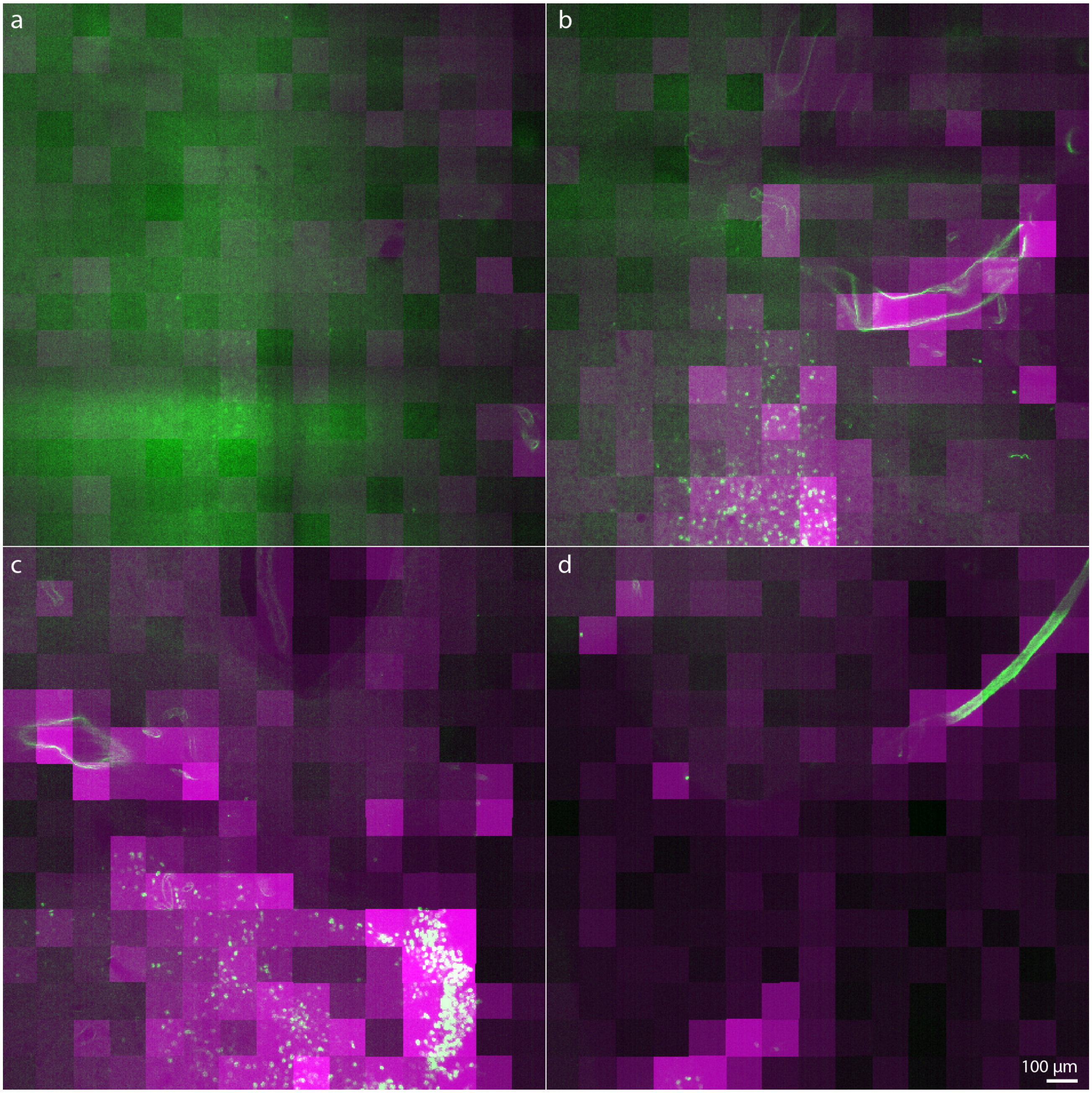
Supplementary Figure 5: *Interactive manual alignment of tiled images.* The BigStitcher GUI offers various ways of manually (pre-)aligning tiled images after import. **(A)** images can be moved to a regular grid with a given tile order and overlap. **(B)** image locations can also be read from a simple *tile configuration* text file. **(C)** selected image(s) can be moved along axes via sliders. **(D)** all changes will be displayed in the BigDataViewer window immediately **(D)** for quick verification.

SUPPLEMENTARY FIGURE 6: Flat-field correction



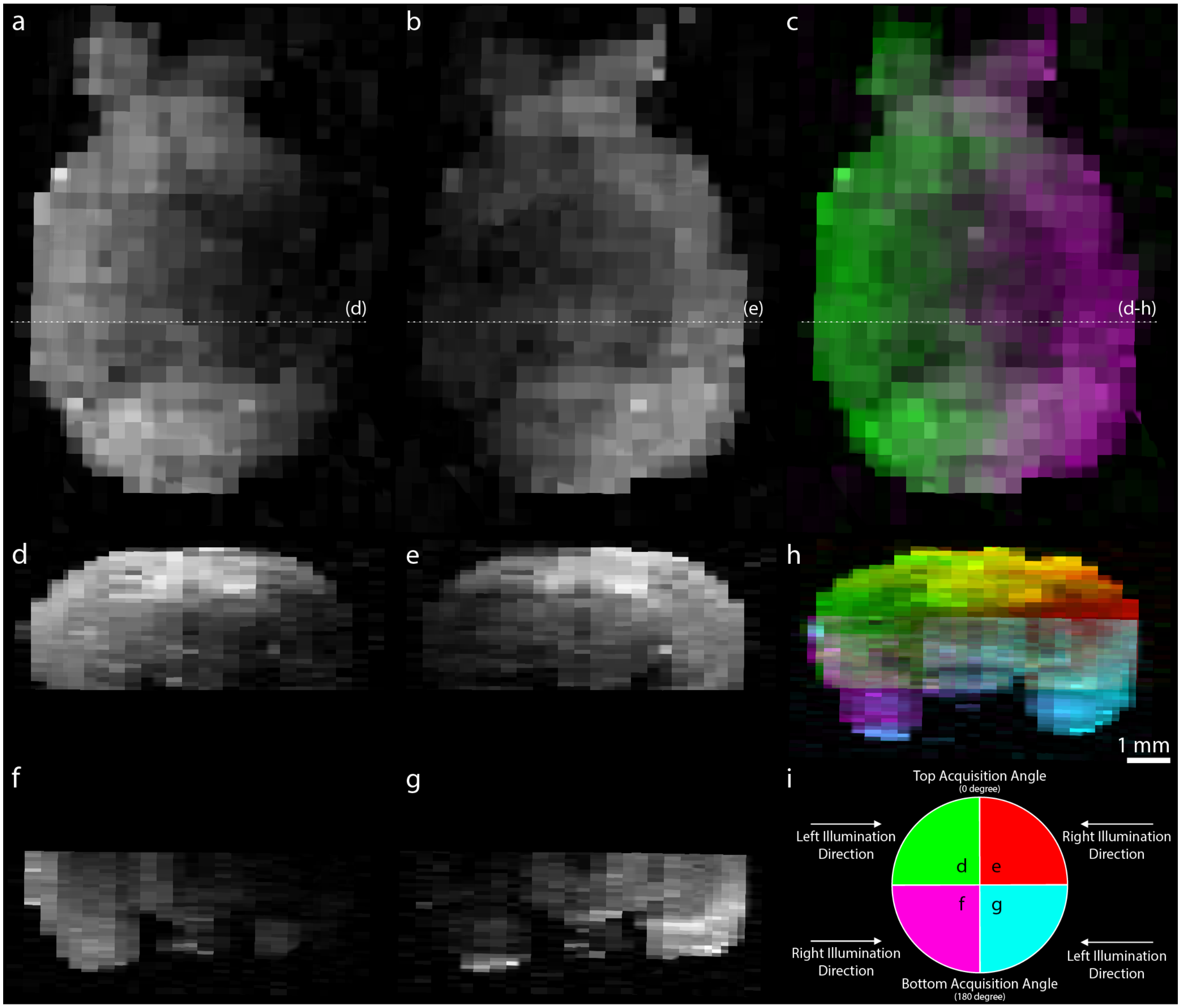
Supplementary Figure 6: *On-the-fly flat-field correction.* The BigStitcher offers correction for camera offsets, fixed pattern noise or uneven illumination. **(A)** Simulation of the effects of a constant background offset and Gaussian illumination/detection efficiency **(C)** on tiled images. By subtracting the *dark image* and modulating with the inverse relative intensity of the *bright image*, such artifacts can be corrected easily **(B)**. The correction is calculated virtually, with optional caching, to allow for immediate inspection of the results. **(A-C)** Flatfield correction as illustrated in this figure is a feature supported by BigStitcher, but has not been applied to any of the datasets shown in this publication.

SUPPLEMENTARY FIGURE 7: Automatic quantification of image quality



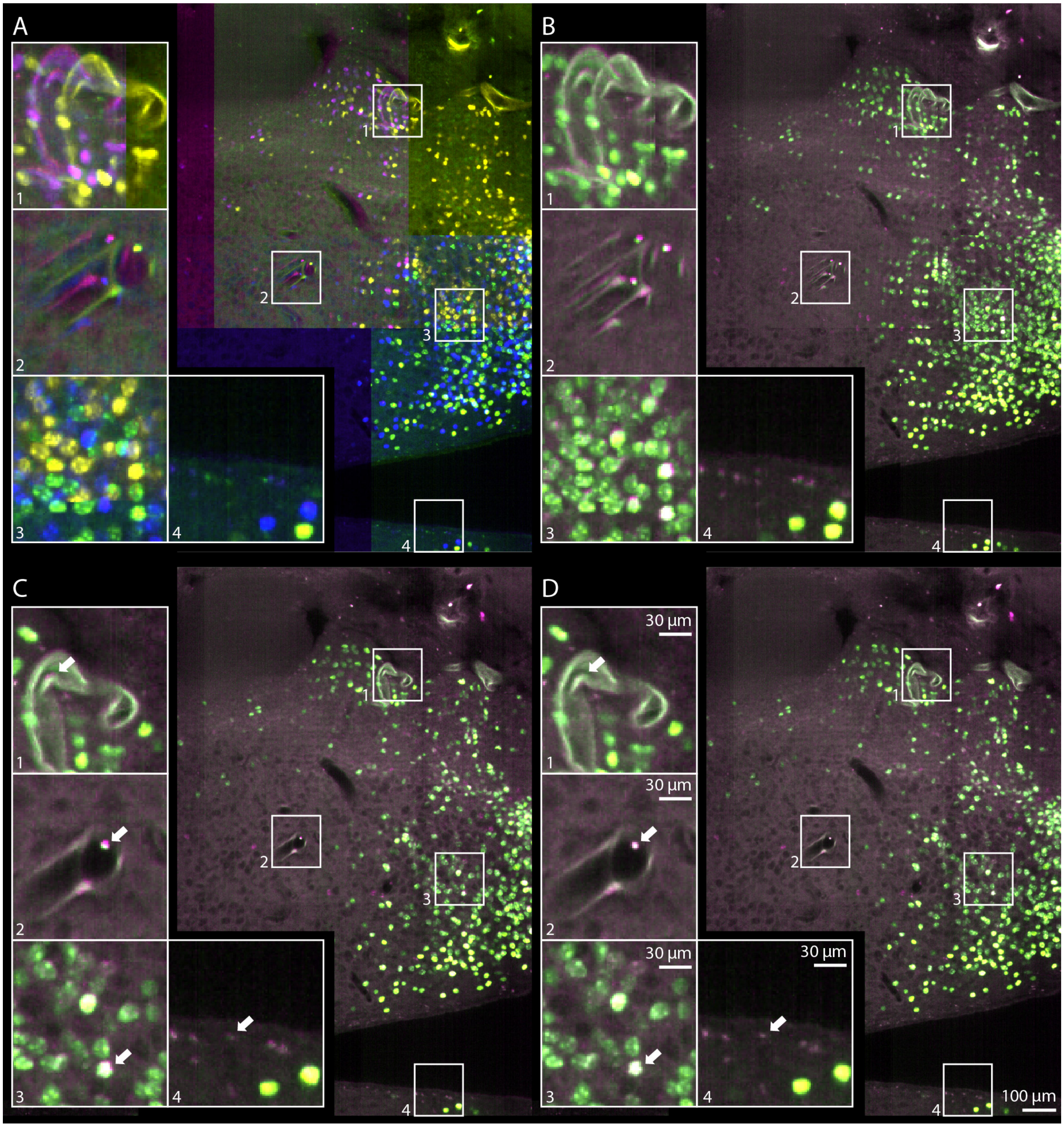
Supplementary Figure 7: *Automatic Quantification of Image Quality.* **(a-d)** Four different z-planes from a volume that overlays the results of the relative Fourier Ring Correlation (rFRC, see **Online Methods**) computed in 128×128 blocks using a spacing of 10 pixels (magenta) and cleared image data (same as **Supplementary Fig. 1**). The rFRC robustly detects areas with high image quality. Note that (a) is deepest inside the tissue and (d) is at the surface of the sample. See **Supplementary Video 8** for an animation of the entire stack. The rFRC was successfully applied to all cleared datasets in this publication (**Supplementary Table 1**), results are also shown in **Supplementary Fig. 1, 8** and **Supplementary Video 8,9**.

SUPPLEMENTARY FIGURE 8: Quality estimation in whole-brain mouse acquisition


a
b
c
(d)
(e)
(d-h)
d
e
h
1 mm
f
g
i
Top Acquisition Angle
(0 degree)
Left Illumination Direction
Right Illumination Direction
d
e
f
g
Right Illumination Direction
Left Illumination Direction
Bottom Acquisition Angle
(180 degree)

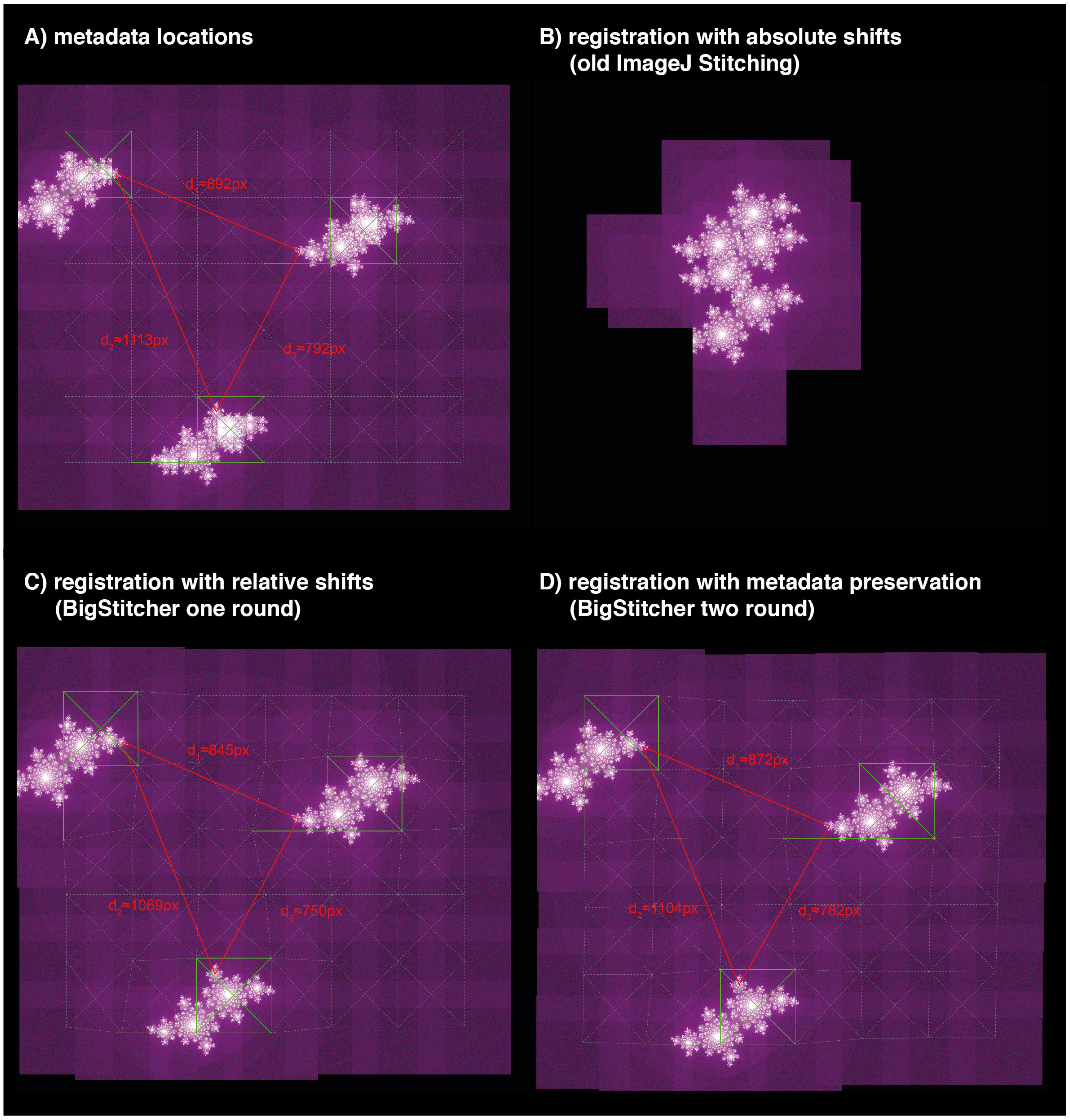
Supplementary Figure 8: *Quality Estimation in Whole-Brain Mouse Acquisition.* Application of our relative Fourier Ring Correlation (rFRC, see **Online Methods**) to the reconstruction of an entire adult mouse brain. The rFRC was computed in 512×512 blocks using a spacing of 256 pixels and transformed and rendered as the reconstructed volume (see Fig. 3d). **(a,b,c)** single slice through rFRC volume based on image data acquired with left illumination (a), right illumination (b), and both overlaid (c), dotted lines outline the orthogonal shown views in (d-h). **(d-h)** orthogonal views to (a-c) highlighting the contribution in image quality from different illumination directions and acquisition angles. **(i)** illustration of the color scheme used in (h) and the type of data displayed in (d-g). See **Supplementary Video 9** for an animation of the entire stack. The rFRC was successfully applied to all cleared datasets in this publication (**Supplementary Table 1**), results are also shown in **Supplementary Fig. 1, 7** and **Supplementary Video 8,9**.

SUPPLEMENTARY FIGURE 9: Affine refinement via ICP



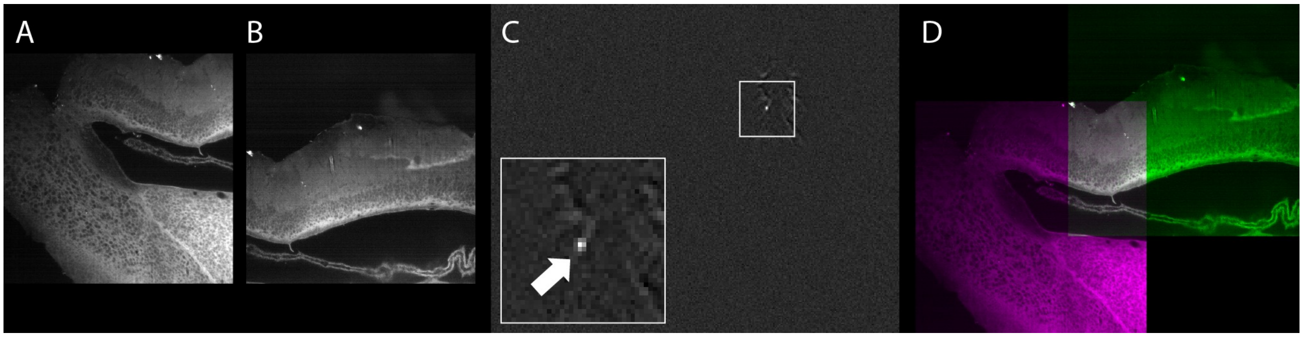
Supplementary Figure 9: *Illustration of different steps for multi-tile alignment* **(A)** Four randomly colored, overlapping image tiles show the typical error when using microscope metadata only. **(B)** Shows the same image tiles as (A), but without random color coding. **(C)** Quality of the registration after applying the phase-correlation based stitching with downsampling 4 and two-round global optimization. **(D)** Result after applying the automatic ICP refinement for tile alignment, spherical and chromatic aberration correction. **(A-D)** Insets highlight specific areas to better appreciate quality differences. **(A-D)** ICP-refinement using affine transformations achieving similar results was applied to all 26 tiles of the dataset, as well as the datasets shown in (**Fig. 1n, 3b-d**).

SUPPLEMENTARY FIGURE 10: Global optimization



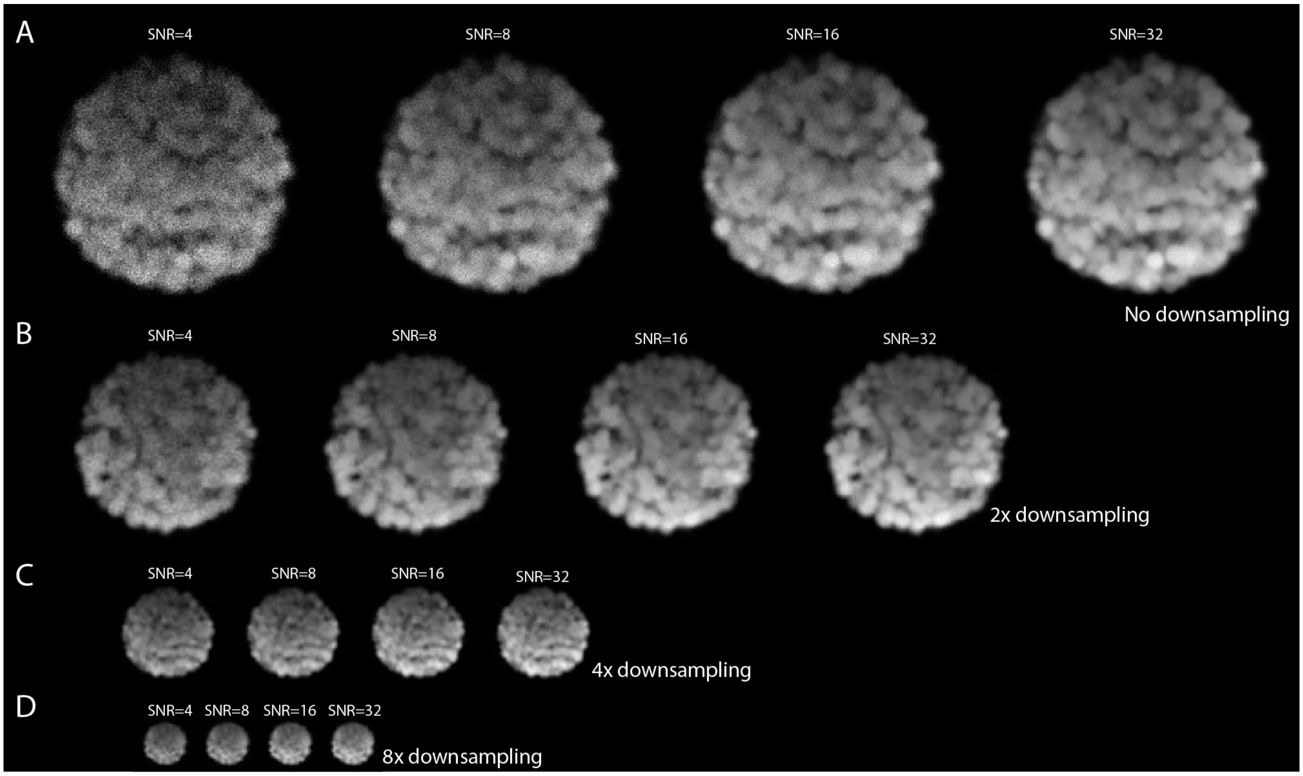
Supplementary Figure 10: *Global optimization of pairwise registration in sparse datasets connected by "empty tiles" (noise only) .* **(A)** Simulation of a tiled image dataset with sparse objects: tiled images of multiple translated Julia fractals moved to a grid according to approximate metadata (with too high overlap). Centers of images for which pairwise shifts can be determined via phase correlation are connected by green lines, whereas centers of neighboring tiles for which no meaningful shift can be calculated are linked by dashed grey lines. Manually measured distances between distinct points in the three fractals are shown in red. **(B)** performing global optimization with *absolute shifts* (as it is done BigStitcher's predecessor, the ImageJ Stitching plugin) will correctly align images within connected components of the link graph but place all fractals close to the origin. **(C)** by using *relative shifts*, BigStitcher will leave disconnected objects at their initial location while still aligning within connected components. **(D)** as registrations are not propagated between unconnected tiles, distances between neighboring objects might change. By running a second round of optimization to align connected components according to metadata shifts and applying the results to the in-component registrations, distances between neighboring objects are preserved as-good-as-possible. **(A-D)** Two-round global optimization as illustrated in this figure is a feature supported by BigStitcher, which has been applied to all datasets used in this publication. Especially the dataset shown in **Fig. 1d,e and Fig. 3d** profits from it since it contains empty tiles.

SUPPLEMENTARY FIGURE 11: Pairwise registration by phase correlation



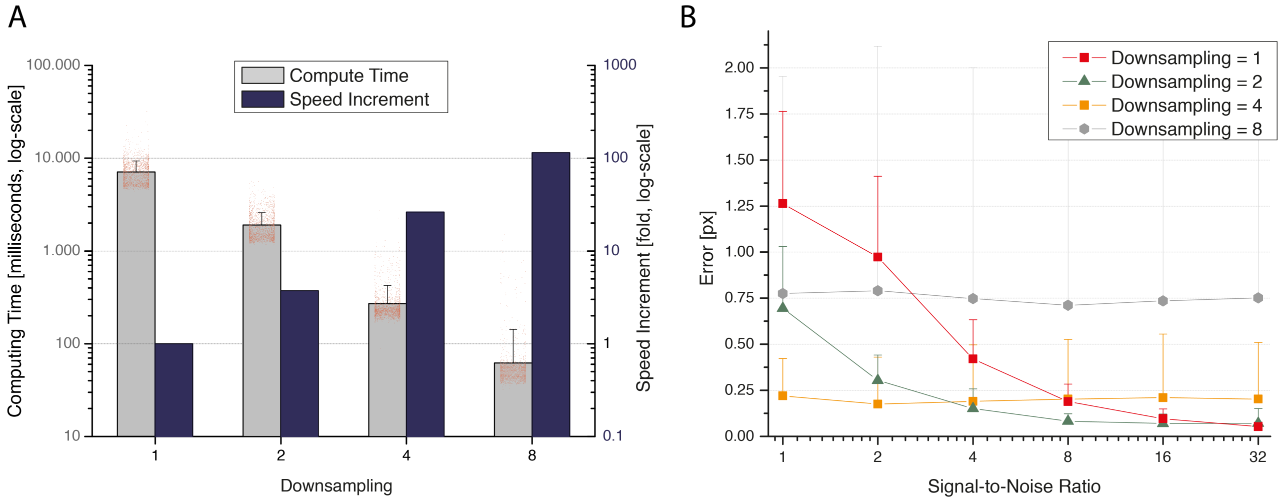
Supplementary Figure 11: *Pairwise registration by phase correlation*. **(A,B)** Central slices of image stacks from a tiled acquisition (non-regular tiling) of a cleared adult mouse hypothalamus. **(C)** Phase correlation matrix (PCM) calculated from the two images shows a single, distinct peak above nearly constant background. The peak location corresponds to the relative translation t of both tiles. **(D)** Central slice through the images aligned according to t, as displayed in interactively during the reconstruction process. **(A-D)** The pairwise registration using phase correlation was used as a first step in the alignment of all cleared and expanded samples used in this publication (**Supplementary Table 1**).

SUPPLEMENTARY FIGURE 12: Downsampling with different SNR



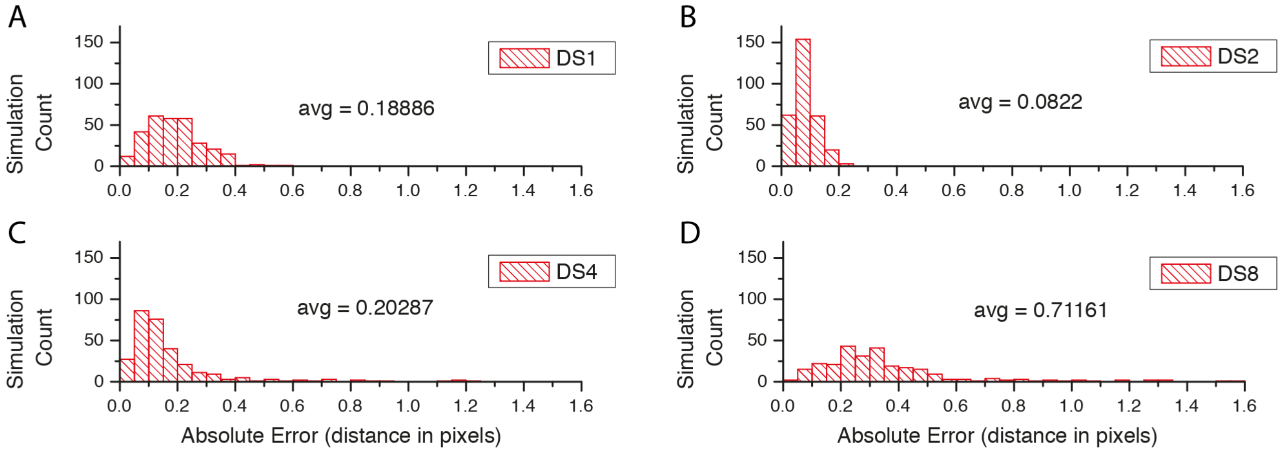
Supplementary Figure 12: *Effects of downsampling on simulated data with different SNR.* **(A)** Simulated image stacks of spheroid-like objects deteriorated by anisotropic sampling, light attenuation, convolution with an anisotropic PSF, and pixel intensity generation using Poisson processes to archive desired signal-to-noise-ratios (SNRs). A central slice through 3d volumes is shown. **(B,C,D)** Effects of downsampling on the simulated images. The effects of Poisson Shot Noise are gradually reduced by the blurring of increasing downsampling. **(A)** For quantification of the alignment quality using these simulations, 300 independent simulations were run for each combination of SNR and downsampling (see **Supp. Fig.** 13 – 15).

SUPPLEMENTARY FIGURE 13: Downsampling statistics 1



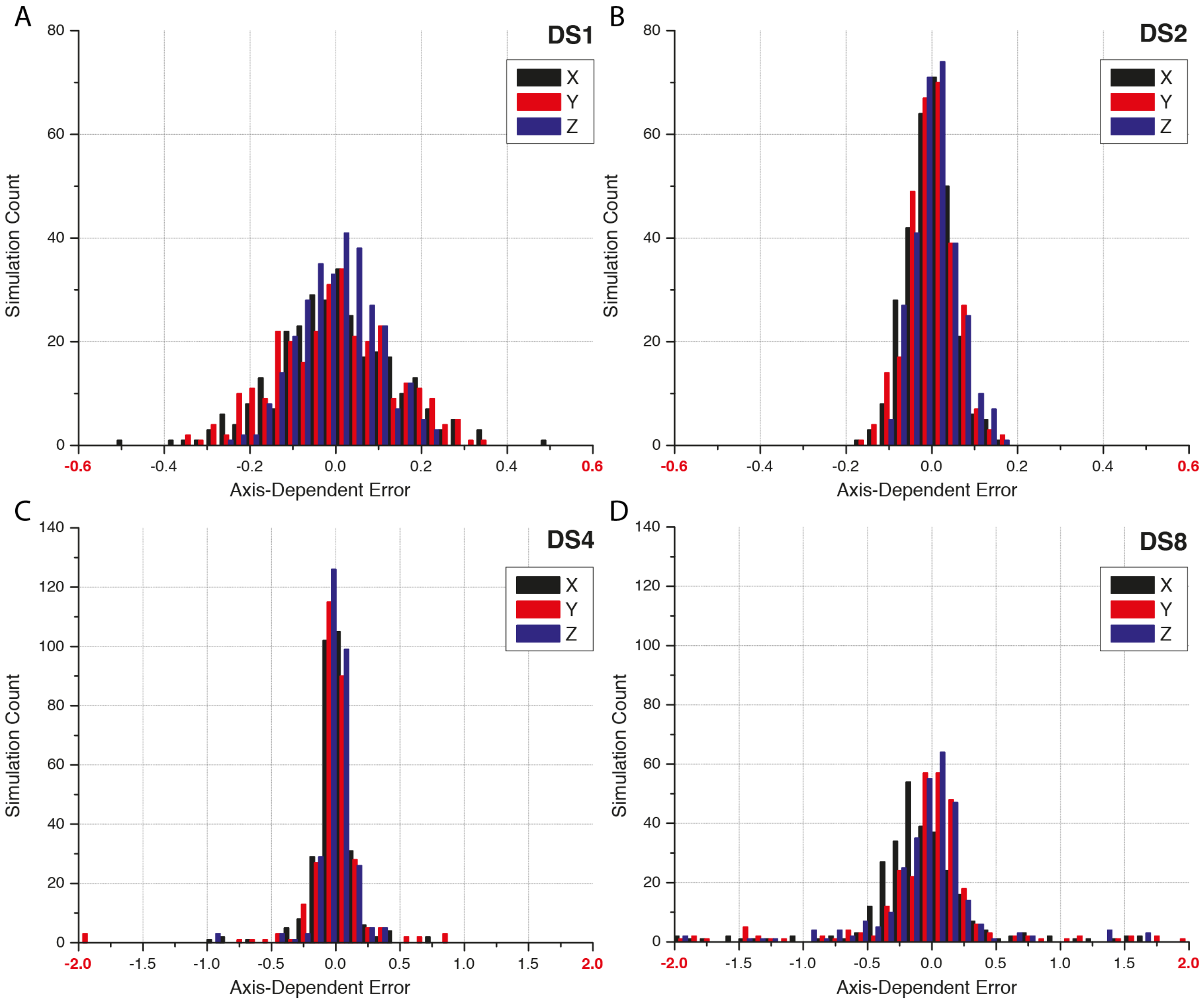
Supplementary Figure 13: *Processing times and overall errors.* **(A)** Processing times for sub-pixel precise identification of overlap between simulation spheroids. With increasing downsampling, the computation time drops significantly. Red dots show individual measurements, note the log-scale. Average (StDev) of computing time is 7122 (2224) msec, 1910 (681) msec, 271 (155) msec, and 62 (80) msec for downsampling 1, 2, 4 and 8, respectively. The speed increments are computed as the ratio of the average compute times, i.e. 1×, ∼4×, ∼26×, and ∼115×, respectively. Compute times were measured in a single thread on a Intel Xeon E5-2640 v4. **(B)** Average errors including their standard deviation for all combinations of SNR and downsampling. **(A,B)** All errors are in units of the input images (no downsampling). For each combination of SNR and downsampling 300 independent simulations were run to compute the values.

SUPPLEMENTARY FIGURE 14: Downsampling statistics 2



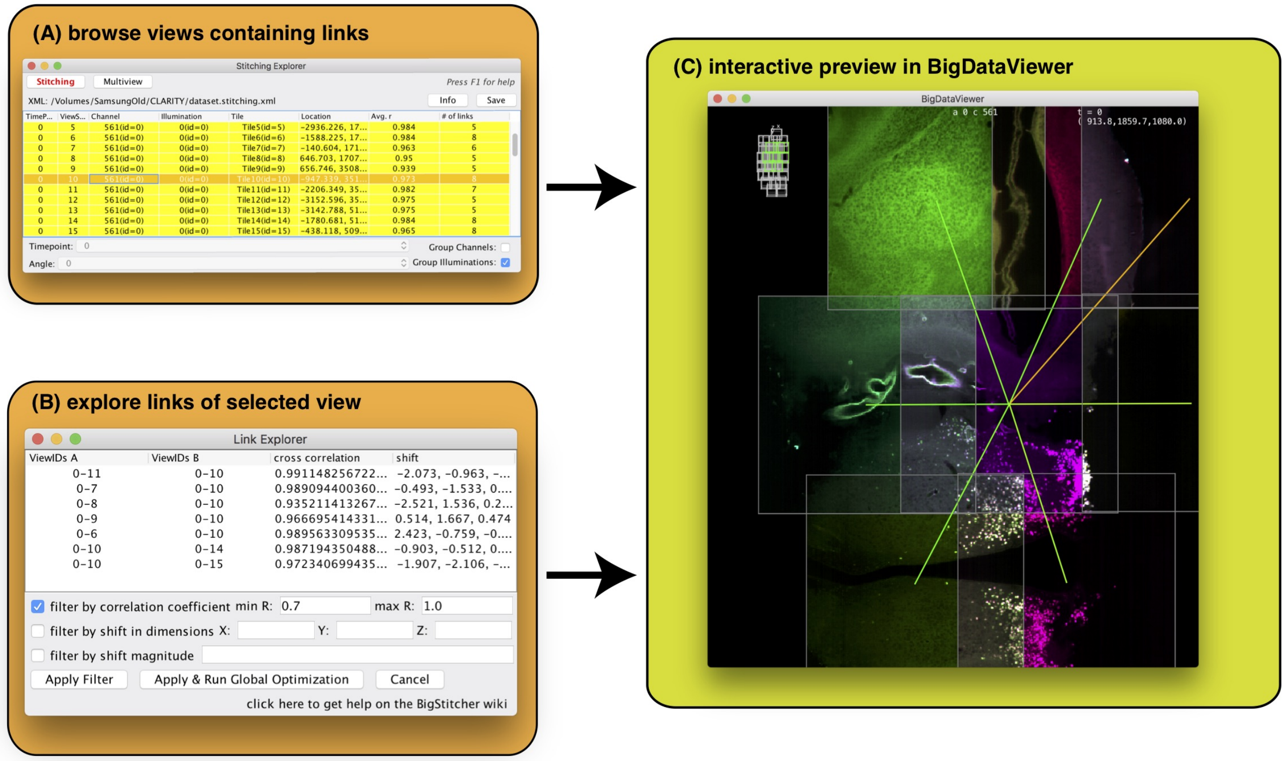
Supplementary Figure 14: *Errors for different downsamplings at SNR=8.* **(A-D)** Histograms showing the distributions of error of the simulations. Errors initially decrease due to the smoothing effect of the downsampling. All errors are in pixel units of the original resolution (DS1). Each histogram consists of 300 independent simulations.

SUPPLEMENTARY FIGURE 15: Downsampling statistics 3



Supplementary Figure 15: *Absolute distance errors at SNR=8.* **(A-D)** Histograms showing the absolute distances between computed and known shift between two simulated spheroids, split by dimension. It illustrates a normal distribution of the error made during the pairwise phase correlation. All errors are in pixel units of the original resolution (DS1). Each histogram consists of 300 independent simulations.

SUPPLEMENTARY FIGURE 16: Interactive inspection and curation of pairwise links



(A) browse views containing links

XML: /Volumes/SamsungOld/CLARITY/dataset.stitching.xml

TimeP...	ViewS...	Channel	Illumination	Tile	Location	Avg. r	# of links
0	5	561(id=0)	0(id=0)	Tile5(id=5)	-2936.226, 17...	0.984	5
0	6	561(id=0)	0(id=0)	Tile6(id=6)	-1588.225, 17...	0.984	8
0	7	561(id=0)	0(id=0)	Tile7(id=7)	-140.604, 171...	0.963	6
0	8	561(id=0)	0(id=0)	Tile8(id=8)	646.703, 1707...	0.95	5
0	9	561(id=0)	0(id=0)	Tile9(id=9)	656.746, 3508...	0.939	5
0	10	561(id=0)	0(id=0)	Tile10(id=10)	-947.339, 351...	0.973	8
0	11	561(id=0)	0(id=0)	Tile11(id=11)	-2206.349, 35...	0.982	7
0	12	561(id=0)	0(id=0)	Tile12(id=12)	-3152.596, 35...	0.975	5
0	13	561(id=0)	0(id=0)	Tile13(id=13)	-3142.788, 51...	0.975	5
0	14	561(id=0)	0(id=0)	Tile14(id=14)	-1780.681, 51...	0.984	8
0	15	561(id=0)	0(id=0)	Tile15(id=15)	-438.118, 509...	0.965	8

Timepoint: 0 Group Channels: □

Angle: 0 Group Illuminations: ☑

(B) explore links of selected view

ViewIDs A	ViewIDs B	cross correlation	shift
0-11	0-10	0.991148256722...	-2.073, -0.963, -...
0-7	0-10	0.989094400360...	-0.493, -1.533, 0....
0-8	0-10	0.935211413267...	-2.521, 1.536, 0.2...
0-9	0-10	0.966695414331...	0.514, 1.667, 0.474
0-6	0-10	0.989563309535...	2.423, -0.759, -0....
0-10	0-14	0.987194350488...	-0.903, -0.512, 0....
0-10	0-15	0.972340699435...	-1.907, -2.106, -...

☑ filter by correlation coefficient min R: 0.7 max R: 1.0

□ filter by shift in dimensions X: Y: Z:

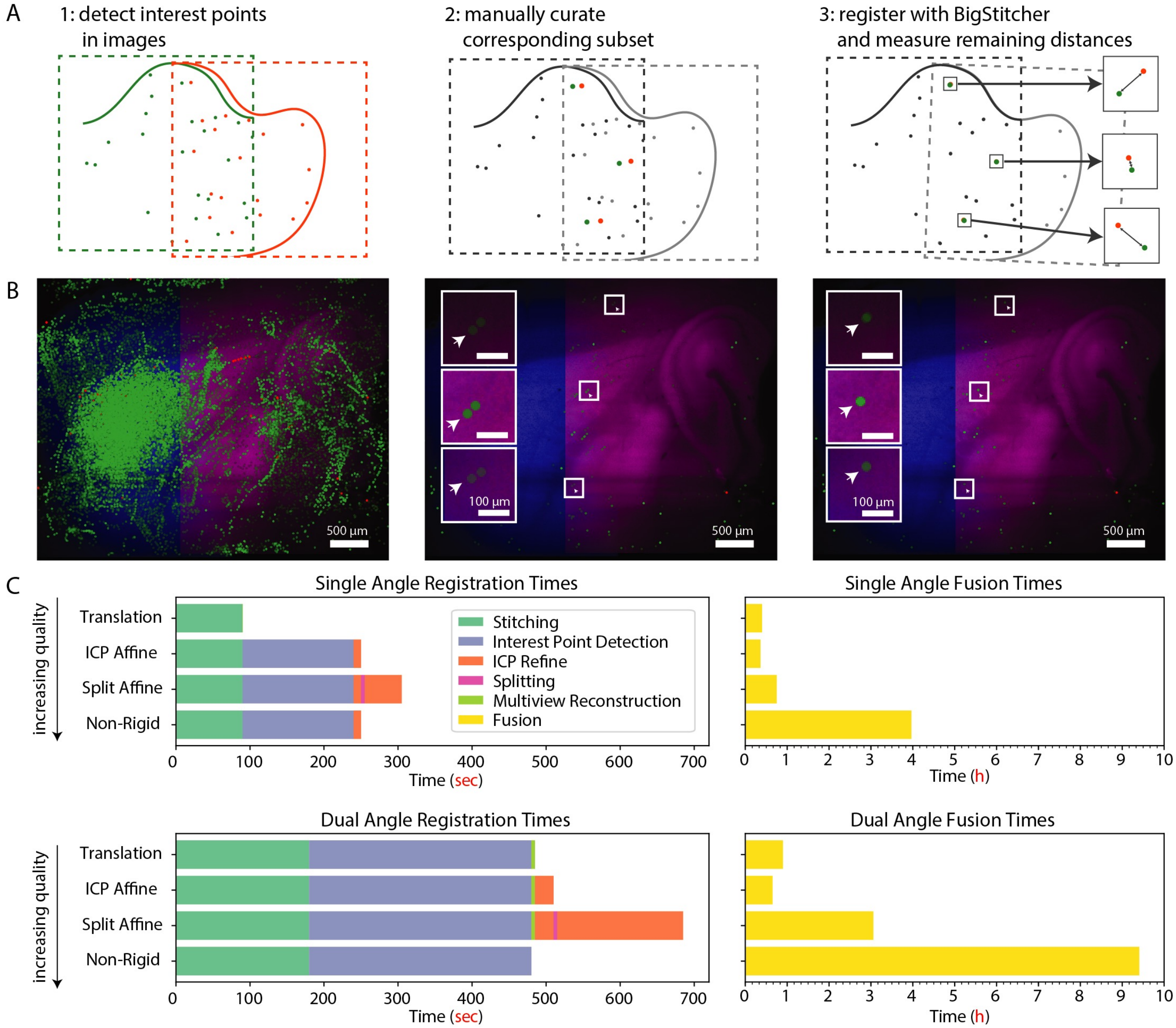
□ filter by shift magnitude

Apply Filter
Apply & Run Global Optimization
Cancel
click here to get help on the BigStitcher wiki

(C) interactive preview in BigDataViewer

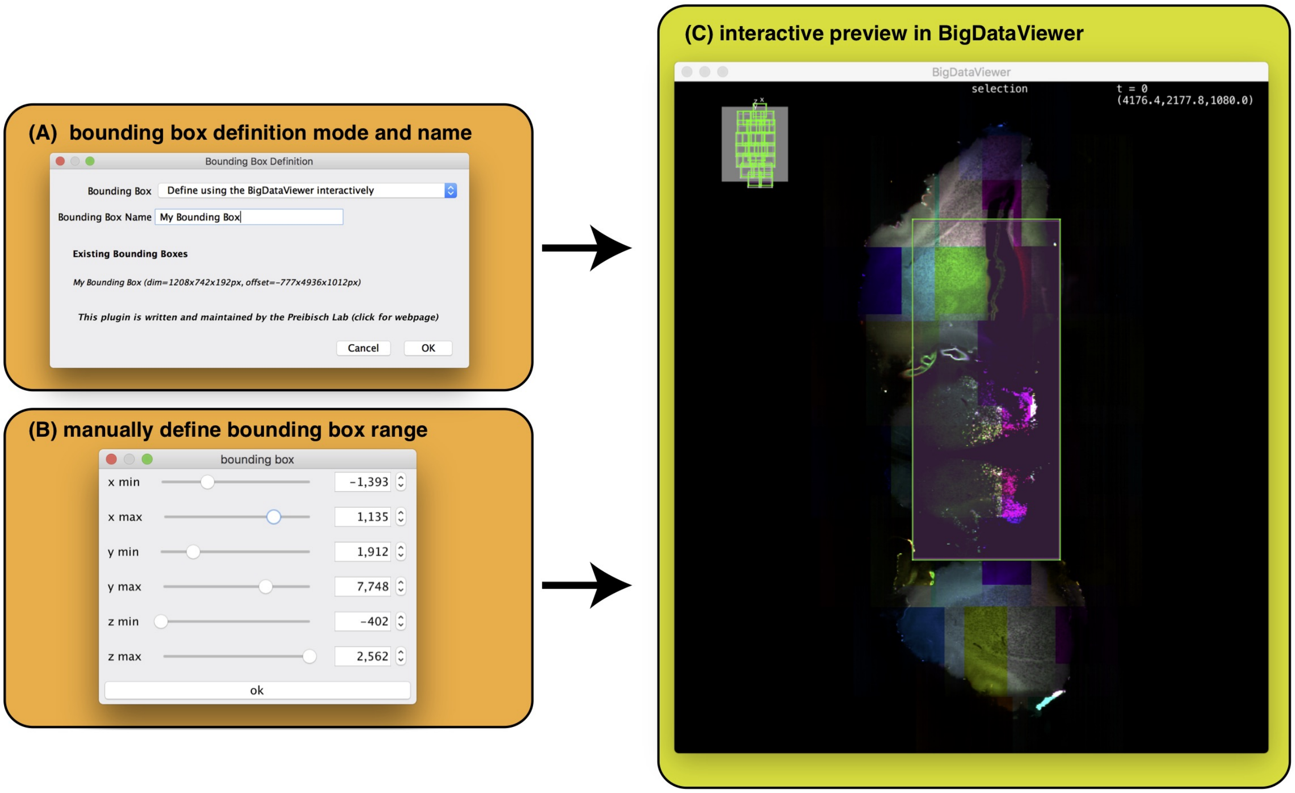
Supplementary Figure 16: *Interactive visualization of links in the link explorer.* The BigStitcher GUI offers to explore and modify calculated links between corresponding tiles in the link explorer menu. **(A)** tiles containing links are displayed in yellow and can be selected. **(B)** display corresponding tiles of the selected view. Single links can be removed manually or through available filtering options. **(C)** corresponding links of the selected view are displayed in real-time in the BigDataViewer.

SUPPLEMENTARY FIGURE 17: Quantification of image registration quality



Supplementary Figure 17: *Quantification of Image Registration Quality.* A multi-view, dual-illumination, multi-tile dataset specifically acquired for verification purposes was used to quantify the registration error (see **Online Methods**, **Fig. 2k-m**, and **Supplementary Fig. 2**). **(A)**: Schematic description of the quantification process for registration accuracy. Interest points are first automatically detected in all images of the dataset (1). Of those, a subset of truly corresponding points interest points was selected for each pair of images (2). After registration with BigStitcher using various transformation models (translation, affine, split-affine, non-rigid), the remaining distance between the manually curated point pairs is used as a measure of registration error (3), actual errors are shown in **Fig. 2m**. **(B)** All interest points detected in two images of the example dataset overlaid on a slice view (left), manually selected corresponding points (middle, note the "doubling" of the paired points as they are not yet aligned, arrows indicate examples) and the same points after registration (right). **(C)** Time required for registration (left) and fusion (right) of the dataset, for a single angle (top) or both angles (bottom). The single angle values are averages of both angles. Fusion was done at full resolution, preserving original data anisotropy. Multi-resolution pyramids of the images were computed beforehand. Processing was done on 2 Intel Xeon E5-2680v4 processors and 256GB RAM, data was loaded from SSDs in RAID0 configuration. This error quantification was performed only on this specifically acquired dataset.

SUPPLEMENTARY FIGURE 18: Bounding-box definition


(A) bounding box definition mode and name
Bounding Box Definition
Cancel
OK
(B) manually define bounding box range
bounding box
ok
(C) interactive preview in BigDataViewer
BigDataViewer
z x y

Bounding Box: Define using the BigDataViewer interactively

Bounding Box Name: My Bounding Box

Existing Bounding Boxes

My Bounding Box (dim=1208x742x192px, offset=-777x4936x1012px)

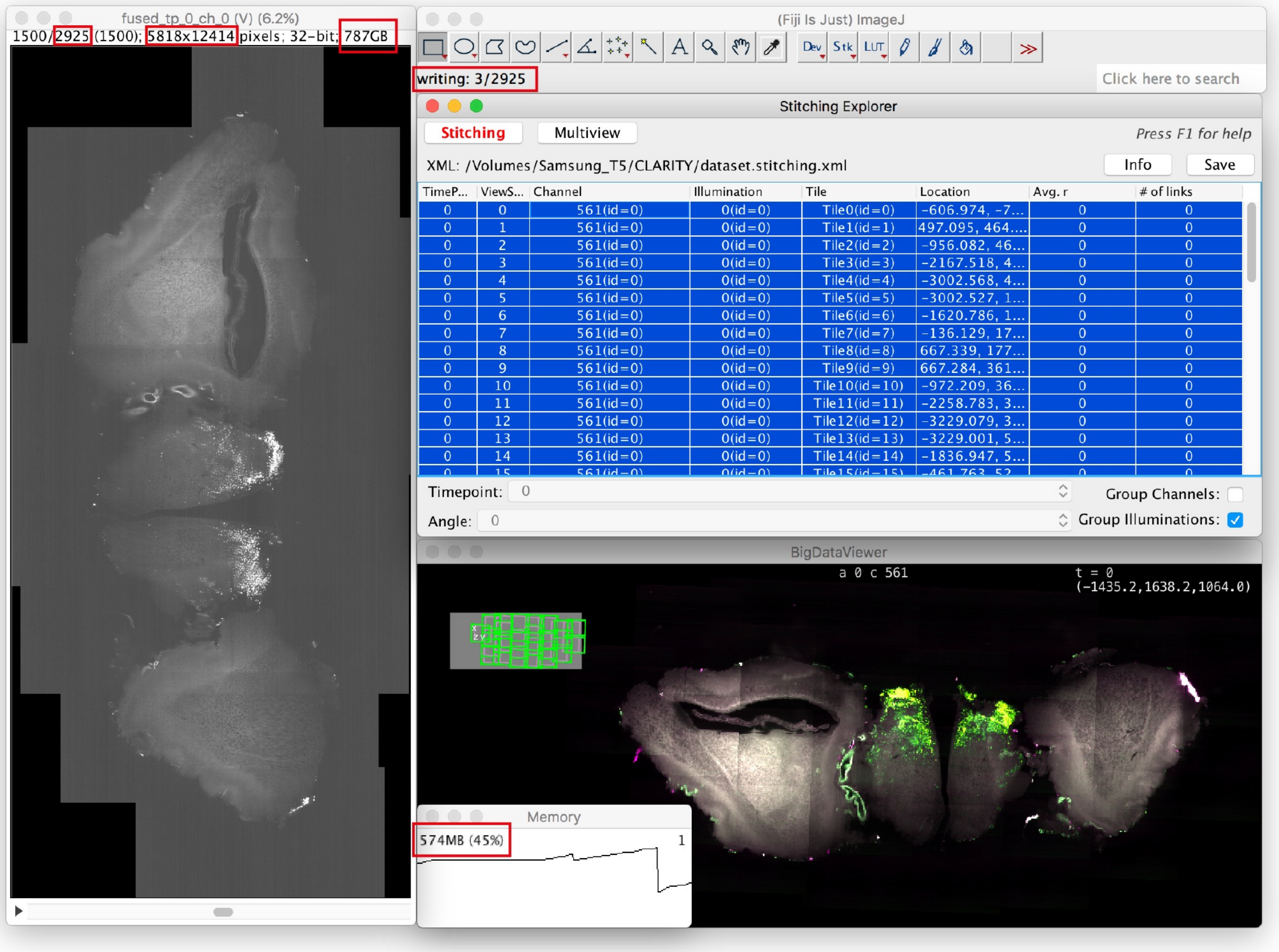
This plugin is written and maintained by the Preibisch Lab (click for webpage)

x min	-1,393
x max	1,135
y min	1,912
y max	7,748
z min	-402
z max	2,562

```
selection
t = 0
(4176.4,2177.8,1080.0)
```

Supplementary Figure 18: *Interactive definition of bounding boxes.* The BigStitcher GUI offers the possibility of defining or modifying regions of interest via the creation of bounding boxes. **(A)** Choose the method used to define a new bounding box. In this case the interactive mode is selected. **(B)** manually define the bounding box range **(C)** Preview the size of the specified bounding box in the BigDataViewer in real-time.

SUPPLEMENTARY FIGURE 19: Virtual fusion of large Image



writing: 3/2925

Stitching Explorer

Stitching | Multiview | *Press F1 for help*

XML: /Volumes/Samsung_T5/CLARITY/dataset.stitching.xml | Info | Save

TimeP...	ViewS...	Channel	Illumination	Tile	Location	Avg. r	# of links
0	0	561(id=0)	0(id=0)	Tile0(id=0)	-606.974, -7...	0	0
0	1	561(id=0)	0(id=0)	Tile1(id=1)	497.095, 464....	0	0
0	2	561(id=0)	0(id=0)	Tile2(id=2)	-956.082, 46...	0	0
0	3	561(id=0)	0(id=0)	Tile3(id=3)	-2167.518, 4...	0	0
0	4	561(id=0)	0(id=0)	Tile4(id=4)	-3002.568, 4...	0	0
0	5	561(id=0)	0(id=0)	Tile5(id=5)	-3002.527, 1...	0	0
0	6	561(id=0)	0(id=0)	Tile6(id=6)	-1620.786, 1...	0	0
0	7	561(id=0)	0(id=0)	Tile7(id=7)	-136.129, 17...	0	0
0	8	561(id=0)	0(id=0)	Tile8(id=8)	667.339, 177...	0	0
0	9	561(id=0)	0(id=0)	Tile9(id=9)	667.284, 361...	0	0
0	10	561(id=0)	0(id=0)	Tile10(id=10)	-972.209, 36...	0	0
0	11	561(id=0)	0(id=0)	Tile11(id=11)	-2258.783, 3...	0	0
0	12	561(id=0)	0(id=0)	Tile12(id=12)	-3229.079, 3...	0	0
0	13	561(id=0)	0(id=0)	Tile13(id=13)	-3229.001, 5...	0	0
0	14	561(id=0)	0(id=0)	Tile14(id=14)	-1836.947, 5...	0	0
0	15	561(id=0)	0(id=0)	Tile15(id=15)	-461.763, 52	0	0

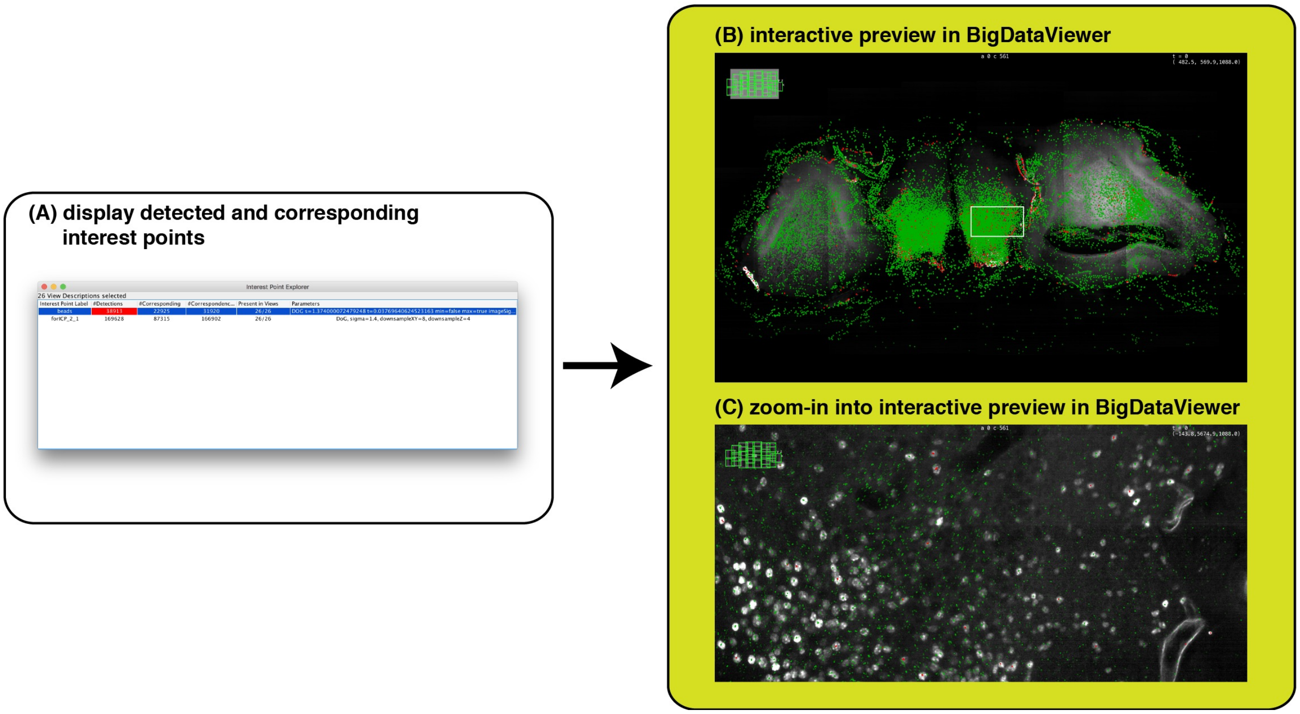
Timepoint: 0 | Group Channels: □

Angle: 0 | Group Illuminations: ☑

574MB (45%)

Supplementary Figure 19: *Virtual Fusion.* Screenshot of a Fiji instance running with **1.25GB of RAM** successfully fusing and saving a **787GB** volume 5818×12414×2925 pixels in size. This is achieved through **virtual fusion** combined with virtual, cached loading of blocked, multi-resolution input images. Red boxes highlight memory consumption, size, and progress. During the fusion process, the BigStitcher and BigDataViewer are interactively accessible.

SUPPLEMENTARY FIGURE 20: Interest point visualization



(A) display detected and corresponding interest points

Interest Point Explorer

26 View Descriptions selected

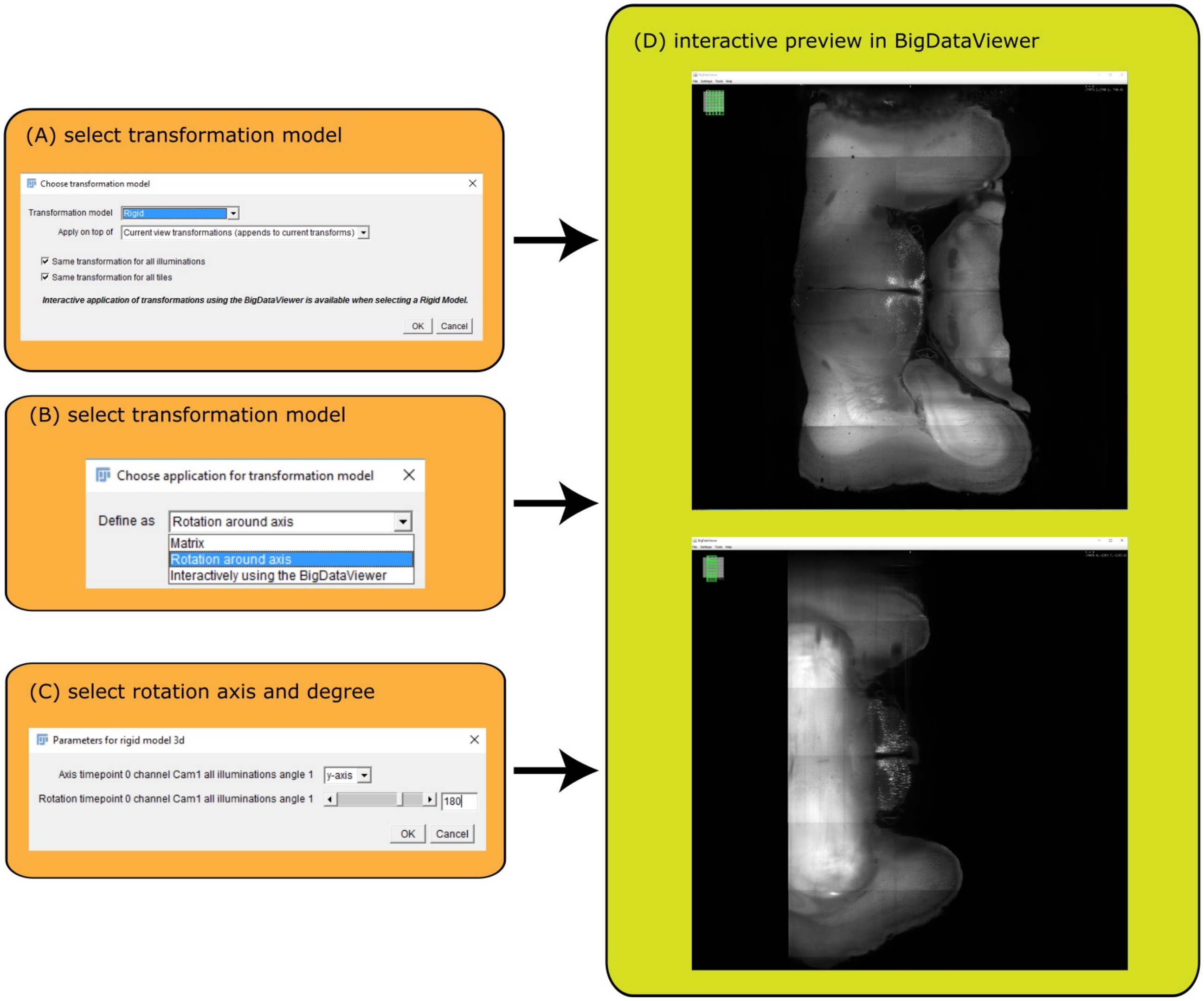
Interest Point Label	#Detections	#Corresponding	#Correspondenc...	Present in Views	Parameters
beads	38913	22925	31920	26/26	DOG s=1.374000072479248 t=0.03769640624523163 min=false max=true imageSig...
forICP_2_1	169628	87315	166902	26/26	DoG, sigma=1.4, downsampleXY=8, downsampleZ=4

(B) interactive preview in BigDataViewer

(C) zoom-in into interactive preview in BigDataViewer

Supplementary Figure 20: *Interactive visualization of interest points.* The interest points explorer allows the visualization of interest points and corresponding interest points between views. **(A)** select desired interest points for visualization. **(B)** preview the interest points overlaid in the BigDataViewer. Red dots intersect with the current image plane, green dots are projections from different z-planes. The white box marks the zoom-in shown in (C). **(C)** Zoom-in into the region outlined in (B).

SUPPLEMENTARY FIGURE 21: Manual transformation of multi-view datasets



(A) select transformation model

Choose transformation model

Transformation model: Rigid

Apply on top of: Current view transformations (appends to current transforms)

☑ Same transformation for all illuminations

☑ Same transformation for all tiles

Interactive application of transformations using the BigDataViewer is available when selecting a Rigid Model.

(B) select transformation model

Choose application for transformation model

Define as: Rotation around axis

(C) select rotation axis and degree

Parameters for rigid model 3d

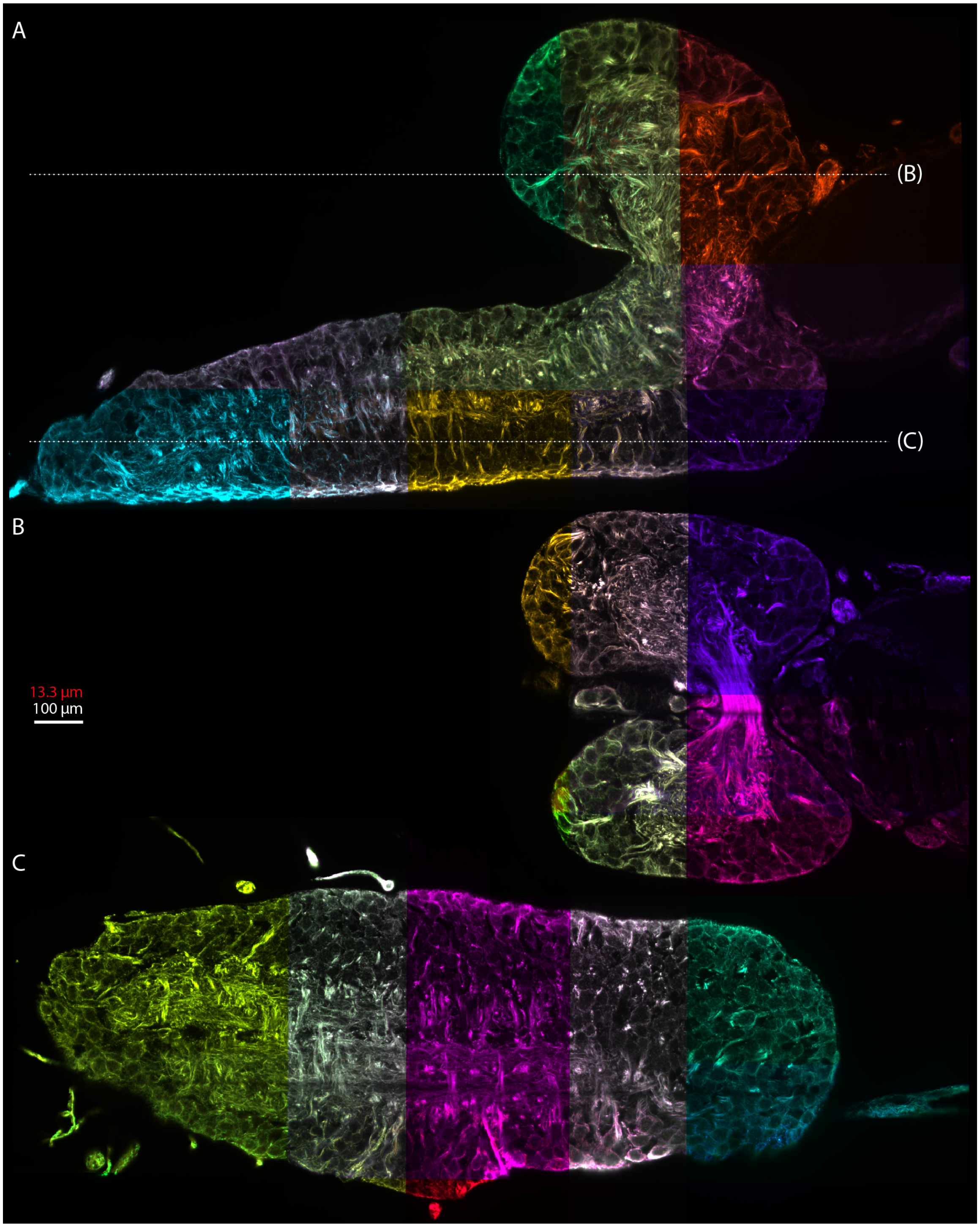
Axis timepoint 0 channel Cam1 all illuminations angle 1: y-axis

Rotation timepoint 0 channel Cam1 all illuminations angle 1: 180

(D) interactive preview in BigDataViewer

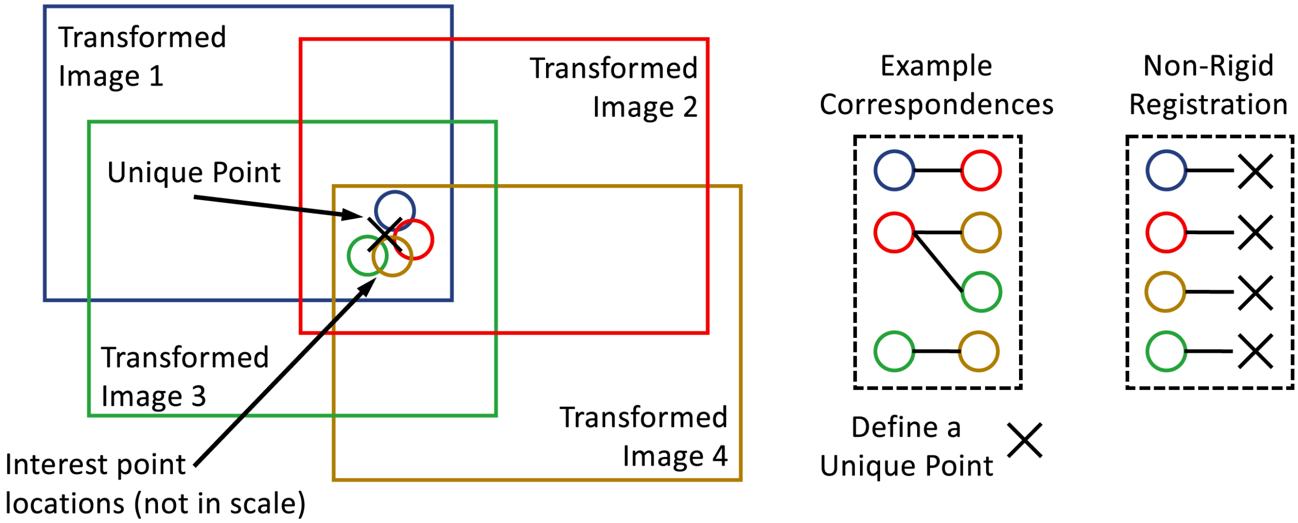
Supplementary Figure 21: *Interactive transformation of views.* Different transformation models can be applied to one or more views and simultaneously visualized in the BigDataViewer. **(A)** Choose transformation model grouping. **(B)** Further define the transformation model. In this case a rotation around the axis is selected. **(C)** Select rotation axis and angle. **(D)** Visualize rotation of the view in the BigDataViewer.

SUPPLEMENTARY FIGURE 22: Expansion microscopy reconstruction



Supplementary Figure 22: *Expansion microscopy stitching* **(A)** All tiles (randomly colored) of one view of the expanded Drosophila central nervous system. Dotted lines highlight orthogonal sections in (B) and (C). **(B,C)** Alignment of an orthogonal view showing at two different cut planes. Red scalebar takes expansion into account. **(A-C)** Expansion microscopy alignment using phase correlation followed by ICP refinement using affine transformations was performed only on this dataset.

SUPPLEMENTARY FIGURE 23: Principles of non-rigid alignment



Supplementary Figure 23: *Principle of Non-Rigid Alignment.* The non-rigid alignment applies a different affine transformation to each pixel of each transformed image. This continuous transformation space is defined by corresponding interest points between overlapping images. We therefore first identify all sets of all corresponding interest points that belong to each other as defined by pairwise correspondences (see Example Correspondences). Each individual set of correspondences across n images then define a $unique\ point$, of which typically hundreds to thousands per image exist. The left part of the figure illustrates a single $unique\ point$, which is defined as the average position of all corresponding interest points. Once all unique points are assigned to each correspondence, the non-rigid transformation can be individually computed for each transformed image.

SUPPLEMENTARY TABLE 1: Summary of all datasets used in this publication.

Dataset	Size	Microscope & Acquisition settings
Coronal slice from a adult mouse brain expressing H2B-eGFP under the neuronal BSX promoter (**Figure 3a + Supplement**)	1920×1920×1039 16 bit stacks 26 tiles from 1 angles (10% overlap) 2 channels, single illumination **0.36 TB total size**	Lightsheet Z.1 with EC Plan-Neofluar 5x/0.16 objective, Depth-of-field: $\sim$25μm 5.37μm LS thickness, 538.3647546μm Confocal parameter 0.915×0.915×2.57425409μm pixels 119.8ms exposure on PCO.edge camera lasers: 561 nm 100%, 488 nm 80% filters: EF1 BP 505-545, EF 2 BP 575-615
Whole adult mouse brain expressing H2B-eGFP in all BSX-expressing neurons (**Figure 1d-n + Figure 2a,f + Figure 3d,e + Supplement**)	1920×1920×770 16 bit stacks 56 tiles from 2 angles (10% overlap) 2 channels, dual illumination **2.2 TB total size**	Lightsheet Z.1 with EC Plan-Neofluar 5x/0.16 objective, Depth-of-field: $\sim$25μm 5.97μm LS thickness, 665.3906758999999μm Confocal parameter 0.915×0.915×4.929649351μm pixels 119.8ms exposure on PCO.edge camera lasers: 561 nm 40%, 488 nm 40% filters: EF1 BP 505-545, EF 2 BP 575-615
	1920×1920×645 16 bit stacks 63 tiles from 2 angles (10% overlap) 2 channels, dual illumination **2.1 TB total size**	Lightsheet Z.1 with EC Plan-Neofluar 5x/0.16 objective, Depth-of-field: $\sim$25μm 10.44μm LS thickness, 2034.834282μm Confocal parameter 0.915×0.915×6.203581395μm pixels 119.8ms exposure on PCO.edge camera lasers: 561 nm 40%, 488 nm 40% filters: EF1 BP 505-545, EF 2 BP 575-615
Coronal slice through an adult mouse brain expressing an H2B-eGFP lineage tracing marker in BSX-expressing neurons (**Figure 3b,e + Supplement**)	1920×1920×828 16 bit stacks 35 tiles from 2 angles (10% overlap) 2 channels, dual illumination **1.55 TB total size**	Lightsheet Z.1 with EC Plan-Neofluar 5x/0.16 objective, Depth-of-field: $\sim$25μm 10.44μm LS thickness, 2034.834282μm Confocal parameter 0.915×0.915×4.930096618μm pixels 119.8ms exposure on PCO.edge camera lasers: 561 nm 50%, 488 nm 50% filters: EF1 BP 505-545, EF 2 BP 575-615
	1920×1920×960 16 bit stacks 35 tiles from 2 angles (10% overlap) 2 channels, dual illumination **1.8 TB total size**	Lightsheet Z.1 with EC Plan-Neofluar 5x/0.16 objective, Depth-of-field: $\sim$25μm 10.44μm LS thickness, 2034.834282μm Confocal parameter 0.915×0.915×4.930916667μm pixels 119.8ms exposure on PCO.edge camera lasers: 561 nm 50%, 488 nm 50% filters: EF1 BP 505-545, EF 2 BP 575-615

Continued on next page

SUPPLEMENTARY TABLE 1: Summary of all datasets used in this publication.

Dataset	Size	Microscope & Acquisition settings
Coronal slice through an adult mouse brain expressing an H2B-eGFP lineage tracing marker in BSX-expressing neurons (zoomed-out) (**Figure 2k-m + Supplement**)	1920×1920×945 16 bit stacks 12 tiles from 2 angles (10% overlap) 1 channel, dual illumination **0.166 TB total size**	Lightsheet Z.1 with EC Plan-Neofluar 5x/0.16 objective (0.5× zoom) Depth-of-field: $\sim$35μm 10.44μm LS thickness, 2034.834282μm Confocal parameter 1.83×1.83×4.93μm pixels 119.8ms exposure on PCO.edge camera lasers: 488 nm 50% filters: EF1 BP 505-545
Whole *C. elegans* during dauer with all neuron nuclei expressing tagRFP in and co-stained with DAPI (**Figure 3e,f**)	750×1920×40 16 bit stacks 16 tiles from 4 angles (10% overlap) 2 channels, single illumination **0.003 (2,96GB) TB total size**	Lightsheet Z.1 with W Plan-Apochromat 20x/1.0 objective (2× zoom) Depth-of-field: $\sim$1μm 1.82μm LS thickness, 56.72238239μm Confocal parameter 0.114×0.114×0.8035μm pixels 59.9ms exposure on PCO.edge camera lasers: DAPI 505 nm 2%, DAPI 561 nm 0.2%, RFP 561 nm 5% filters: DAPI CAM BS Mirror, RFP CAM BS SBS LP 560, RFP EF 1 BP 575-615
Central nervous system of a Drosophila 1st instar larva with immunostaining for tubulin (**Figure 3c,e + Supplement**)	2048×2048×923 stacks 26 tiles from 2 angles (28% overlap) 1 channels, dual illumination **0.188 TB total size**	IsoView with SpecialOptics 16x/NA 0.71 objective Depth-of-field: $\sim$2μm 4.92μm LS thickness, 416.0μm Confocal parameter 0.4125×0.4125×0.8125μm pixels 20.1ms exposure on Orca Flash 4.0 v2 sCMOS camera lasers: 488 nm 10.6mW filters: BP488/10 EX, BP525/50 EM

Note: Depth-of-field was estimated based on the descriptions on this resource Depth of Field and Depth of Focus — MicroscopyU.

SUPPLEMENTARY TABLE 2: Performance comparison

Performance comparison						
Data Size	130 Mb / 63 Gb / 300 Gb	130 Mb / 63 Gb / 300 Gb	130 Mb / 63 Gb / 300 Gb	130 Mb / 63 Gb / 300 Gb	130 Mb / 63 Gb / 300 Gb	1.67 Tb
Software	**Illumination Selection**	**Stitching** (requires Illumination Selection)	**ICP refinement**	**Fusion** 8x down-sampling full resolution (requires Illumination Selection)	**Virtual Fusion** @ full resolution	**Multi-view Registration**
BigStitcher	n.a. / 9 s / 14 s	4 s / 1 min 28 s / 7 min 15 s	9 s / 57 s / 2 min 52 s	1 s / 9 s / 44 s 8 s / 1 h 19 min / 8 h 43 min	1 s / 15 s / 19 s	6 min 30 s
TeraStitcher	X	16 s / 34 min 43 s / 2 h 29 min	X	1 min 15 s / 14 min 24 s* / 1 h 4* 2 min 29 s / 26 min 41 s* / 2 h 35 min*	X	X
ImageJ Stitcher	X	10 s / 61 min / 5 h 35 min	X	X / X / X 16 s / 10 h 36 min* / 44 h 40 min*	X	X
Xuv Tools	X	2 s / X / X	X	X / X / X 9 s / X / X	X	X
Arivis	X	27 s / 47 h 15 min / 7 d 4 h*	X	X / X / X 2 s / 5 min* / 51 min*	X	X

BigStitcher benchmark for processing a terabyte-sized multi-view dataset						
Data Size	1.67 Tb					
Software	**Illumination Selection**	**Stitching**	**ICP refinement**	**Fusion** @ 4x downsampling	**Virtual Fusion** @ full res. (display / save)	**Multi-view Registration**
BigStitcher	59 s	1h 14 min	7 min 28 s	1 h 1 min	56 s / 23 h 50 min*	6 min 30 s

Supplementary Table 2 *Comparison of BigSticher features and performance with other available stitching programs for four different datasets with different sizes (130 Mb, 63 Gb, 300 Gb, and 1,67 Tb).* Benchmarks were performed on a HP Z840 workstation running Windows 10 Pro with two Intel Xenon E5-2667v4 CPUs and 512 GB memory. The benchmarks for the Arivis software were performed on a HIVE system running Windows Server 2012 R2 with two Intel Xenon E5-2640v3 CPUs and 256 GB memory. The latest stable version of each stitching program was used. BigStitcher datasets were stitched using 4× times (x,y) downsampling, and fusion was performed at stated downsampling levels. Correctness of the stitching could only be confirmed in the BigStitcher due to the flexibility of interactive inspection. Processing of multi-view, dual-illumination datasets as well as ICP refinement and virtual fusion is only possible in BigStitcher. Note that fusion in BigStitcher also performs intensity adjustment. All displayed values are averaged from three independent runs of each respective software except those marked with a black star. **Note that image fusion results are not comparable since all other applications fuse datasets using translation-only, which is a significantly simpler problem that cannot align the datasets sufficiently well while BigStitcher uses affine models. Therefore translation-only results on dual-illumination datasets are grayed and marked with a red star.** A cross indicates that the functionality is not supported by the software, 'n.a.' indicates that the dataset did not require this feature.

SUPPLEMENTARY NOTES

1. Sample clearing

Clearing of brain tissue was performed using the CLARITY protocol.[1] Mice were deeply anesthetized by intraperitoneal injection of 100 mg/kg Ketamine and 15 mg/kg Xylazine. Mice were exsanguinated by transcardial perfusion with 25 ml cold PBS followed by whole body perfusion with 25 ml cold monomer solution (4% v/v acrylamide, 4% w/v Paraformaldehyde (PFA), 0.25% w/v VA-044 in PBS). The brains were collected and fixed in monomer solution for 2 more days. Next, the whole brains were placed in fresh monomer solution and oxygen was removed from the tubes by vacuum and flushing the tube with nitrogen gas for 15 minutes. The samples were then polymerized by placing the tubes in a 37°C water bath under gentle shaking for 2 hours. Polymerized brains were placed in clearing solution (4% SDS in 200 mM Boric acid). Brains were incubated in clearing solution for 1 week at 37°C with daily solution change. Then, the brains were actively cleared using the X-Clarity setup from Logos Bioscience for 24 hours with a current of 1 A at 37°C. Cleared brains were washed twice overnight with 0.1% v/v Triton X-100 in PBS and once with PBS.

2. Expansion microscopy (ExM)

Expansion microscopy sample preparation

For Expansion Microscopy (ExM), the nervous system of a 1st instar Drosophila larva of was extracted, fixed in 4% PFA/1xPBS/0.1%Triton for 1 hour and washed 2x10 min in 1xPBS/0.1% Triton. Before each antibody usage, the nervous system and the antibodies were blocked in 5% goat serum/1xPBS/0.1% Triton for one hour. Following the blocking, the nervous system was incubated overnight at 4°C in 1:500 monoclonal Anti-α-Tubulin antibody produced in mouse (Sigma Aldrich T6199 1mg/ml). After 5x10 min washing (1xPBS/0.1% Triton), the secondary antibody 1:250 Anti-Mouse CF$^{\text{TM}}$488A antibody produced in goat (Sigma Aldrich AB4600387 2mg/ml) was added overnight at 4°C.

Detailed expansion microscopy protocol

Acryloyl-X, SE (6-((acryloyl)amino)hexanoic acid, succinimidyl ester; here abbreviated AcX; Thermo-Fisher) was resuspended in anhydrous DMSO at a concentration of 10 mg/mL, aliquoted and stored frozen in a desiccated environment. AcX stock solution was diluted in 1xPBS to a final concentration of 0.1 mg/mL AcX. Specimens were incubated in this 0.1mg/mL AcX solution for $>$6 h, at RT. Monomer solution (1xPBS, 1M NaCl, 1.84M sodium acrylate, 0.35M acrylamide, 3.2mM N,N'-methylenebisacrylamide) was mixed, frozen in aliquots, thawed fully, vortexed, and cooled to 4°C before use. Concentrated stocks of the initiator ammonium persulfate (APS, 10% w/w), accelerator tetramethylethylenediamine (TEMED, 10% v/w) and inhibitor 4-hydroxy-2,2,6,6-tetramethylpiperidin-1-oxyl (4-HT, 0.5% w/w) were prepared as concentrated stock solutions, which were frozen in aliquots and then fully thawed and vortexed before use. Initiator, accelerator and inhibitor stock solutions were added to the monomer solution at a ratio of 2uL each per 94uL monomer solution to produce complete monomer solution. Specimens were washed 2x15min in complete monomer solution, on ice with shaking. Specimens were transferred to 3D-printed gelation chambers sized 1cm x 1cm and 0.3mm deep, along with 30uL of complete monomer solution. Chamber was covered with a cover glass and transferred to a humidified 37°C incubator for 2hr for gelation and gel curing.

Proteinase K (New England Biolabs) was diluted 1:100 to 8 units/mL in digestion buffer (50 mM Tris (pH 8), 1 mM EDTA, 0.5% Triton X-100, 1 M NaCl) to produce proteinase solution. Gel was recovered from chamber and incubated fully immersed in proteinase solution overnight at RT, with shaking. The digested gel was next incubated in at least a 10-fold excess volume of monomer solution with accelerator and inhibitor (no initiator) 2x15min, followed by complete monomer solution 2x15min on ice, with shaking. (Initiator is omitted from the first two washes to prevent premature gelation.) During incubation, a glass slide and cover glass are coated with parafilm by laying parafilm with paper backing onto the glass surface (parafilm down) and scraping a razor blade across the backing, then removing the backing. A 5mL syringe filled with silicone grease was used to apply four dabs (approx. 50uL each) of grease to the glass slide, at the corners of a rectangle slightly smaller than the cover glass. The gel was transferred to the coated slide and excess fluid was removed. The cover glass was placed over the gel, parafilm side down, contacting the dabs of grease. The cover glass was gently pressed down, squeezing the grease, until the coverglass contacted and sat flat across the gel. Excess complete monomer solution was backfilled into the resulting chamber to impede access of atmospheric oxygen to the gel. The completed chamber was moved to the 37°C humidified incubator for 2hr for gelation and curing.

The resulting doubly-gelled specimen was recovered from the chamber, and excess gel was trimmed away. The trimmed double gel was washed in excess volumes of doubly deionized water for 0.25-2 h to expand. This step was repeated 3-5 times in fresh water, until the size of the expanding sample plateaued.

3. Imaging strategies

3D images of cleared mouse brains were imaged using the Zeiss Lightsheet Z.1 microscope. Each sample was attached to the sample holder using a cyanoacrylate-based glue. The mounted sample was placed in the FocusClear pre-filled imaging chamber. Images were acquired using the EC Plan-NEOFLUAR 5×/NA 0.16 objective together with the LSFM 5×/NA 0.1 illumination objectives on a Zeiss Light-sheet Z.1. The data was acquired using dual side illumination and from different angles. Images were collected with two 1920×1920 pixel sCMOS cameras and stored in the Zeiss CZI file format.

Fixed C. elegans dauer larvae were embedded in 1.2% agarose containing fluorescent beads and imaged using the same microscope in a water-filled sample chamber. Imaging was performed using the 20×/NA 1.0 objective with additional 2× zoom.

3D images from a cleared and expanded central nervous system of a Drosophila 1st instar larva were acquired using an IsoView light-sheet microscope[2] that has been modified for multi-tile acquisition. To prepare the sample for imaging, excess gel surrounding the expanded sample was removed using a scalpel, leaving four flat and smooth gel surfaces for imaging. Some extra gel was left underneath the sample for mounting, and the sample was affixed to a cylindrical post using a cyanoacrylate-based glue. The mounted sample was placed in the imaging chamber filled with deionized water. Orthogonal views for each tile of the sample were acquired sequentially by switching the illumination and detection orders in IsoView. Images were acquired using SpecialOptics 16×/NA 0.71 objectives and recorded using full frames (2048×2048 pixels, pixel pitch of 0.4125 μm in sample space) of Orca Flash 4.0 v2 sCMOS cameras. The sample was held stationary during multi-view acquisition of each tile, and depth-sectioned images were acquired every 0.8125 μm by translating the detection piezos over a range of 750 μm. A tile for each view thus covered a field of 832 μm (X), 832 μm (Y), and 750 μm (Z). Automated tiling across the entire sample was achieved by moving the sample in predefined steps of 600 μm in X, Y, and Z until full coverage was achieved. Bi-directional light-sheet illumination was achieved using opposing SpecialOptics objectives and the illumination NA was chosen to be 0.0315 for a confocal parameter of approximately 416 μm. The light-sheets from opposing arms were shifted approximately by their Rayleigh length (208 μm) toward the illumination objectives so that each light-sheet provided uniform coverage of the respective half in the acquired image.

A summary of the most important acquisition parameters can be found in **Supplementary Table 1**.

4. Data import

Import of data

Microscopy acquisitions are saved in a multitude of vendor-specific formats, custom formats, and general formats such as TIFF stacks. We developed an extendable, user-friendly interface that automatically imports almost any format and extracts relevant metadata such as illumination directions, sample rotation, and approximate image positions using Bioformats.[3] The assignment of attributes to the image stacks in the raw data is usually automatic, or can be achieved with minimal interaction from the users. Therefore, the importer supports interactive placement of image tiles using regular grids or text file-based definitions (**Supplementary Fig. 5**). BigStitcher accesses image data through memory-cached, virtual loading,[4] optionally combined with virtual flat-field correction (**Supplementary Fig. 6** and **Supplementary Note 5**). Performance is optimal when images are stored using a multiresolution, blocked, compressed format enabling interactive visualization, processing and interaction with terabyte-sized image datasets. The importer therefore suggests by default to resave single-block images (e.g. TIFF) into the BigDataViewer HDF5 format.[5] Alternatively, by making use of cached ImgLib2 data structures, we support virtual loading of image planes from the raw files including caching of already loaded planes.

Data import is described in detail on the BigStitcher Wiki `https://imagej.net/BigStitcher_Define_new_dataset`. We additionally added an example youtube video that illustrates how the most generic import from TIFF stacks works in BigStitcher: `https://youtu.be/aUofNP6V0lg`. In case direct import from a custom format fails, we therefore suggest to manually re-save data as TIFF stacks and subsequently importing them into BigStitcher. It is important to preserve the calibration of the image stacks in the process.

`SpimData` data format

We internally represent our image data and metadata using an extended version of the `SpimData` data format of BigDataViewer.[5] Each image stack is defined by a (`ViewSetup`, `TimePoint`)-combination. We extend the format by giving each `ViewSetup` the following *attributes*: `Channel` to represent color channels, `Illumination` to represent illumination directions, `Angle` to represent multi-view acquisition angles and finally `Tile`, representing (local) x,y points in a multipoint acquisition.

In addition to those attributes, we store detected interest points, bounding boxes (named sub-volumes in which we can *fuse* or deconvolve images), point spread functions for deconvolution and pairwise registrations (that have yet to be used in global optimization) for each (`ViewSetup`, `TimePoint`) *view*. For each image stack, we also store its *registration* (i.e. the transformation from pixel to world coordinates) as a list of affine transform matrices. The registration steps described below will typically prepend another transformation matrix to this list. Finally, the `SpimData` is associated with an `ImgLoader` object that can make image pixel data available as an ImgLib2[4] `RandomAccessibleInterval` given a (`ViewSetup`, `TimePoint`) *view id*.

The `SpimData` data structure can be saved as an XML *project file*, allowing users to manually edit it with any text editor. We automatically save previous versions of the project file to provide the user with the ability to un-do registration steps.

5. Flat-field correction

Flat-field correction is the process of correcting for image artifacts due to uneven illumination or detection efficiency or fixed-pattern noise. Aside from being visually unpleasing, especially in tiled acquisitions, these artifacts can also effect image registration and downstream quantitative image analyses. We therefore offer simple on-the-fly correction for a *dark image* (which might be nonzero due to e.g. camera offset) and a *bright image* (representing uneven illumination or detection efficiency across the field-of-view). We calculate corrected pixel intensities C from a raw image R and bright and dark images B and D as:

$$C_x = \frac{(R_x - D_{x'}) * \overline{(B - D)}}{(B_{x'} - D_{x'})} \tag{1}$$

The correction images can either have the same dimensionality as the raw images, in which case $x' = x$, or have lower dimensionality (e.g. when using 2D correction images on a 3D image stack), in which case $x' = (x_1 \dots x_n)$ with n being the dimensionality of the correction images. If a dark image is not provided by the user, we assume it to have constant intensity of 0 (corresponding to no background offset). Likewise, if no bright image is provided, we assume it to have constant intensity of 1 (uniform illumination and detection efficiency).

We implemented the flat-field correction as a wrapper around an `ImgLoader`, calculating corrected pixel intensity values on-the-fly (with optional caching) every time an image is loaded. That way, the corrected images are available for all other processing steps such as intensity-based registration, interest point detection or image fusion, but it is still possible to activate or de-activate the correction or change bright or dark images after the initial flat-field correction. A separate (bright, dark)-correction image pair can be set for every image in the dataset by modifying the XML project file, while in the GUI we offer user-friendly assignment of correction images to every (channel, illumination direction)-pair.

6. Pairwise shift calculation

In BigStitcher, we currently offer three ways of calculating shifts between a pair of images: the Fourier-based *phase correlation* algorithm, the Gradient-descent-based *Lucas-Kanade* algorithm, both intensity-based methods, as well as interest point-based alignment.

Phase correlation

By default, we calculate pairwise translational shifts of two images I_1 and I_2 using phase correlation[6,7] using our new ImgLib2 implementation.[4] In noiseless images, the method produces a phase correlation matrix (PCM) Q containing a single δ-impulse at the location corresponding to the shift between the two images. Real images might contain multiple peaks (**Supplementary Fig. 11**) and we localize the n highest peaks in Q by detecting peaks with subpixel accuracy using a n-dimensional implementation of a quadratic fit.[8] Aside from allowing subpixel-accurate registration, we can use the precision obtained from the subpixel accuracy of the phase correlation to counteract the effects of downsampling (**Supplementary Fig. 12**), allowing us to achieve registration of similar quality to full-resolution with significant performance gains (**Supplementary Fig. 13–15**). Due to the periodic nature of the Fourier shift theorem, each peak in the PCM actually correspond to 2^n possible shifts in n dimensions. We therefore test each of these candidate shifts by calculating the cross-correlation between the images I_1 and I_2, optionally with interpolation in the case of sub-pixel shifts.[8] We choose the shift vector t corresponding to the highest cross correlation as the final result after applying downsampling correction, if necessary.

It is often necessary to not only align two single images but groups of images, e.g. all channels of a tile. We therefore implemented a flexible framework for the registration of grouped images (see below). The two images I_1 and I_2 can have arbitrary affine pre-registrations such as sample rotation, correction of axial scaling, or already performed registration steps. If pre-registrations of I_1 and I_2 are identical, or are only based on different translations or axis-aligned scalings, we run the phase correlation on (downsampled) raw input images, otherwise on virtually fused images (**Supplementary Note 16**).

Lucas-Kanade

In addition to the default phase correlation-based pairwise shift calculation, we offer registration via an ImgLib2 implementation of the *inverse compositional* formulation of the gradient descent-based Lucas-Kanade optical flow algorithm.[9] While the algorithm is applicable to a variety of transformation models, we currently stick to estimating a translation vector t. If the pairwise registration converges, we calculate the cross correlation of the overlapping portions of the images as a quality metric for the pairwise registration.

Intensity-based registration of grouped images

In many use cases, one might want to align not single images but groups of images, e.g. all channels of a tile, in the pairwise registration step. For this, we implemented a flexible framework for the registration of grouped images.

Each attribute of the images can be set to be an *axis of application*, an *axis of comparison* or an *axis of grouping*. The registration will proceed by first splitting the images by the application attributes, i.e. grouping all images that have the same value for these attributes. In each group, the images are then split by the comparison attributes and finally, the remaining image groups (that differ only in the grouping attributes) are combined into one image stack by either averaging all images for each grouping attribute or picking the image with a specific instance of the attribute.

In a typical application, the stitching of tiled datasets, we would, for example, start by *applying* the registration to all (`Angle`, `TimePoint`)-combinations individually, *comparing* by `Tiles` and finally *grouping* by `Illumination` and `Channel` for each tile, e.g. by averaging illumination directions and picking a specific channel.

Intensity-based registration of images with pre-registrations

The two images I_1 and I_2 can have arbitrary pre-registrations, i.e. pixel coordinates x_{px} are mapped to world coordinates x_w via the affine transforms $x_{w,I_1} = A_{I_1} x_{px,I_1} + b_{I_1}$ and $x_{w,I_2} = A_{I_2} x_{px,I_2} + b_{I_2}$. Depending on the values of A_{I_1} and A_{I_2}, we consider two cases: If they are equal, i.e. the pre-registrations differ only by a translation, we perform the shift calculation on the raw pixel data of the overlapping volume to get a shift vector t for I_2 in pixel coordinates. The transformation in world coordinates is then given by $R\left(\begin{smallmatrix} I & & t \\ 0 & \cdots\ 0 & 1 \end{smallmatrix}\right)R^{-1}$ with $R = \left(\begin{smallmatrix} A_{I_2} & & b_{I_2} \\ 0 & \cdots\ 0 & 1 \end{smallmatrix}\right)$. If the pre-registrations differ in more than just translation, we create virtually transformed images of the smallest rectangular bounding box enclosing the overlapping volume and use them as input to the registration. As the virtual input images are already in world coordinates in this case, the resulting transformation matrix for I_2 is simply $\left(\begin{smallmatrix} I & & t \\ 0 & \cdots\ 0 & 1 \end{smallmatrix}\right)$

Interest-point based

For interest-point based pairwise registration, we detect local extrema in either Difference-of-Gaussian or Difference-of-mean filtered images, optionally followed by subpixel refinement of the detections via a quadratic fit.[8] If we are registering a pair of image *groups*, the interest points of each image in the group are pooled, with optional replacement of point clusters within a user-defined radius by their center.

For each image, we apply the current (affine) registrations to the pixel-coordinate interest points and then determine *candidate point matches* via descriptor matching.[10,11] We then perform model-based outlier removal via the RANSAC algorithm,[12] yielding a set of *inlier point pairs*, $C_{inliers}$, and an optimal translation t for I_2, minimizing $\sum_{(ip_1,ip_2)\in C_{inliers}} ||ip_1 - ip_2 - t||^2$

7. Quantification of pairwise image stitching using downsampling

To assess the effect of downsampling on the pairwise stitching we use simulations of spheroid-like objects at different signal-to-noise ratios (SNRs) as ground truth. We create realistic images by mimicking image creation in light-sheet microscopy including optical sectioning, 3-fold anisotropy between xy and z, light attenuation, convolution, and pixel intensity generation using Poisson processes.[13] Importantly, pairs of overlapping images that we use for benchmarking the subpixel phase correlation method are created using different Poisson processes and are additionally rendered with half a pixel offset of the full resolution images to avoid nearly identical overlaps at high SNRs due to the simulation process (**Supplementary Fig. 12**).

We simulate 300 pairwise overlaps, each at SNRs ranging from 1 to 32, and lateral downsamplings ranging from 1× to 8×, where axial downsampling is matched as good as possible to achieve near-isotropic resolution as in the actual software. We illustrate that across SNRs downsampled images yield a constant registration quality, which even exceeds that of registration at full resolution for low SNRs. This is achieved through a combination of the smoothing effect during downsampling (**Supplementary Fig. 12**) and precise subpixel-localization (**Supplementary Fig. 13–15**). Due to the smoothing effect, registration quality therefore initially increases at 2-fold and 4-fold downsampling (**Supplementary Fig. 13–15**), while when using more downsampling, the loss of pixel resolution outweighs the effect of smoothing and hence the quality drops. Registrations with a constant quality of an average error of below one pixel can be computed at a fraction of the computing time compared to full resolution, typically 4 - 115 times faster. Existing outliers are filtered during global optimization and overall registration quality can further be improved during the ICP[14]refinement step.

Simulated data was created using the $net.preibisch.stitcher.headless.StitchingPairwise$ class in the BigStitcher package (release version 0.3.3). Since it is a Maven artifact, the versions of all dependencies are defined and the corresponding version can be built automatically from that source code state (`https://github.com/PreibischLab/BigStitcher/commit/0d7f79a59ab15fb1805157ab72c5bc9802b02fbd`).

8. MultiView Registration

For MultiView registration, i.e. alignment of image taken from different angles (or also time series stabilization), we first detect interest points in the individual images as described above (6). Images may be grouped (and are by default if we are, e.g. registering tiled acquisitions from multiple angles for which we already aligned the tiles via an intensity-based method) according to their attributes, by pooling their interest points and optionally merging clusters of interest points.

Pairwise point correspondences can either be established by geometric local descriptor matching, a modified version of the iterative closest point (ICP[14]) algorithm or by simply matching the center of mass of the point clouds of both images (note that in this case the registration will be constrained to be a translation). Using the link graph (V, C) and pairwise point correspondences P_{ij} established thus, we calculate the final registration by performing global optimization as described above (12), optionally with iterative link removal and a second round to preserve metadata.

Geometric Local Descriptor Matching

To identify corresponding interest points in between two point clouds, geometric local descriptor matching combined with random sample consensus (RANSAC[12]) has been proven to be a powerful technique.[10,11] The basic idea to express each interest point as a geometric constellation using its n (typically three) nearest neighboring interest points. The vector difference between two descriptors then describes how similar the local area of two points is. A geometric local descriptor (GLD) is assumed to be a correspondence candidate if it is at least m (typically one to ten) times more similar than the second most similar GLD.[8] True corresponding interest points between two point clouds are finally identified using RANSAC on a regularized affine transformation model. However, fast GLD matching using the rotation-invariant technique based on geometric hashing[10] requires relatively randomly distributed points to robustly identify correspondences, while the non-accelerated, redundant, translation-invariant counterpart[11] identifies correspondences reliably in non-rotated point clouds of only up to a few thousand points in reasonable time. Here, we developed a new matching procedure by extending both techniques to better suit the requirements when attempting to identify corresponding interest point in between point clouds of prior unknow size derived from imaged structures that are potentially rotated relative to each other.

Redundancy is a powerful mechanism for GLD matching. It uses additional nearest neighbors but excludes some of them sequentially during matching making it more robust to potentially mis-detected interest points.[11] We therefore extend the fast rotation-invariant technique based on geometric hashing[10] with the capability for redundancy. This significantly increases the chance of being able to align randomly oriented point clouds very fast, albeit at low inlier ratios (ratio of true correspondences to total number of correspondence candidates). Rotation invariance is not desired if both point clouds are known to be approximately in same orientation, for example if the rotation of the sample performed by the microscope was known and has been applied to the dataset. Checking for potential rotations simply increases the chance for wrong correspondence candidates. We therefore implemented a fast translation-invariant GLD based on geometric hashing that supports redundancy. All four versions of GLD are available in BigStitcher to enable robust multi-view alignment.

9. Quantification of overall registration quality

To quantify registration quality, we acquired an as-small-as-possible (166GB), cleared section of an adult mouse brain. It is imaged at lower magnification from two angles (0° and 180°) and in a 2×3 tile configuration with dual-sided illumination for each angle (**Fig. 2l,m**, **Supplementary Fig. 2,17** and **Supplementary Table 1**).

We identified a ground-truth set of corresponding interest points in directly adjacent images by manually selecting bright spots from a set of interest points that were automatically detected using Difference-of-Gaussian filtering and subpixel-accurate local maxima determination (Supplementary Fig. 17a,b). For each image pair, we selected between 19 and 52 corresponding points, in total 692.

We then registered the dataset in BigStitcher for tiled acquisition only, tiled acquisition across illumination directions, and for the multi-tile, dual illumination, multi-view case. This is achieved by grouping the images either by angle and illumination direction, just by angle, or not at all. For the single-view cases, we performed translational alignment by stitching the images using phase correlation. For an all-to-all registration with a translation model, the images of angle 2 were manually rotated by 180° and then all images were aligned using interest points by fast translation-invariant GLD matching followed by RANSAC[12] and global optimization using a translation model. All translation-model alignments were refined using ICP[14] as described above. The point correspondences determined during ICP were further used for non-rigid refinement.

For virtual re-blocking, each original image was split into 2×2×2 sub-blocks (with 120px overlap in xy and 100px overlap in z). After the re-blocking, 4–28 manually selected point correspondences remained between each set of directly adjacent blocks. For each of the image groupings and registration models used, we calculated an average error of the manually selected point correspondences:

$$e_{avg} = \frac{1}{|I|}\sum_{i_1 \in I}\left(\frac{1}{|C(i_1)|}\sum_{i_2 \in C(i_1)}\left(\frac{1}{|PM(i_1,i_2)|}\sum_{(p_1,p_2)\in PM(i_1,i_2)} ||T^{i_1}(p_1) - T^{i_2}(p_2)||^2\right)\right) \tag{2}$$

With I being the set of images, $C(i)$ the adjacent images of an image i (ignoring diagonal pairs for which no corresponding interest points were manually selected as well as pairs that are not in the same group, e.g. when grouping by angle and illumination direction, and pairs from the same original image in the virtually blocked dataset), $PM(i,j)$ the corresponding manually selected interest points of images i, j and T_i the transformation of image i.

To estimate the lowest theoretically achievable errors given a certain transformation, we use only the manually selected point correspondences to calculate a globally optimal registration (and optionally the non-rigid refinement thereof) of the images and then calculate the average error from the same point correspondences as described above. In the virtually blocked case, we also use manually selected point correspondences (826 in total) between adjacent blocks within the same original image for the registration (but ignore them for the final error calculation).

Despite relatively small aberrations in this sample as compared to entire mouse brains (compare with **Fig. 2b,c,d** and **Fig. 2k**) we illustrate that using only translation as transformation model is only reasonable for tiled acquisitions that do not include multiple illuminations or multiple acquisition angles, yet even there spherical aberrations persist that question the standard use of translation models in general (**Supplementary Fig. 4,9**). The alignment errors increase when aligning across illumination directions and greatly increase when aligning different acquisition angles. Importantly, please note that the alignment quality across different illumination directions is significantly reduced on larger samples when using only translation models (compare **Fig. 2b,c,d** and **Fig. 2k**). Using the affine, split-affine or non-rigid registration functionality, BigStitcher can sharply reduce the registration errors in large cleared and expanded samples. As a trade-off between speed and quality we usually choose affine or split-affine registrations.

10. Multiview deconvolution

In addition to real-time image fusion, we offer deconvolution of bounding-box-defined volumes using a multi-view formulation of the iterative Richardson-Lucy deconvolution algorithm[15,16] with Tikhonov regularization[17] and various optimizations.[13] The PSFs required for deconvolution can be extracted from interest points detected in the images (e.g. when subdiffraction fluorescent beads were incorporated with the sample, see section 11) or supplied as TIFF stacks with odd dimensions by the user. BigStitcher offers GPU acceleration of the deconvolution on CUDA-capable Nvidia GPUs.

To allow deconvolution of multi-tile views, we extended the original deconvolution[13] to be based on the virtual fusion. Thereby, any number of input image tiles are virtually fused and serve as one of input views for the multi-view deconvolution. Proper multi-view deconvolution of partly overlapping samples requires sophisticated weight normalization in between views,[13] which we implemented to be computed virtually. Since also the input views are also virtually loaded, the memory requirement of the deconvolution solely depends on the output image size and shows a significantly increased memory-efficiency. All virtual inputs and weights are additionally cached, ensuring highest-possible processing performance for systems with large amounts of RAM.

11. PSF measurement and PSF extraction

In light-sheet microscopy, measured PSFs often differ significantly from simulated ones due to variable precision of light-sheet alignment in every experiment. Therefore, light-sheet deconvolution usually relies on the extraction of PSFs from the actual experiment19,25. To be able to perform PSF extraction in cleared tissue we developed a new protocol. Estapor Fluorescent Microspheres (F-XC 030) were diluted 1:20000 in monomer solution containing bis-acrylamide (0,05% v/v bis-acrylamide, 4% v/v acrylamide, 4% w/v Paraformaldehyde (PFA), 0.25% w/v VA-044 in PBS). The monomer solution was polymerized under constant vacuum and shaking at 37°C for 2 hours. The formed hydrogel was incubated in FocusClear overnight and imaged using the Zeiss Lightsheet Z.1 microscope with the same experimental settings used to acquire previous samples. For C. elegans dauer imaging fixed larvae were embedded in 1.2% agarose together with Estapor Fluorescent Microspheres (F-Z 030), diluted 1:2000. For ExM data acquired on the IsoView microscope depth-sectioned images (0.4125 μm step) of fluorescent beads (200nm diameter) embedded in 0.6% low-melting-temperature agarose were imaged using the same experimental settings as for sample imaging. For all samples, PSFs were extracted by detecting interest points in the acquired bead images. Potential bead aggregates were excluded by manual removal on the maximum intensity projection using the BigStitcher module “Manage Interest Points > Remove Interactively”.

12. Global optimization

Estimation of globally optimal transformations

The pairwise registration step results in *links* between image (groups) V (note that since we do not use the actual image *content* in the global optimization, we will refer to the images by their integer *id* in this section: $V \subset \mathbb{N}$). The links can be either in the form of pairwise transformations T^p (such that coordinates x from two images V_i and V_j can be transformed according to $T^p_{ij}(x_j) = x_i$) or *point correspondences* PM from which such transformations can be estimated. The pairwise registrations thus form a *link graph* (V, C) with edges $C = \{(i,j) \in V \times V | T^p_{ij} \in T^p\}$ between image pairs for which we could determine pairwise transformations. Simply traversing a spanning tree of the link graph and propagating the pairwise transformations can lead to the compounding of pairwise registration errors, even if the traversal is done along a *minimal* spanning tree determined according to some quality metric q_{ij}, e.g. cross-correlation, of the pairwise registrations.

We thus make use of an algorithm for globally optimal registration by iterative minimization of square displacement of point correspondences[18,19] for reaching a reasonable consensus in this case. This point match-based framework allows for flexible grouping and fixing of images, is applicable to, among others, time series-, chromatic channel- or view-registration and can easily be adapted to incorporate the pairwise transformations from e.g. phase correlation. The algorithm is agnostic of the transformation model (e.g. translation, affine transform,...), with the only requirement being that the model parameters can be estimated by a least-squares fit from point correspondences.

We determine the globally optimal registrations R given the image (groups) V, pairwise links C, pairwise n-dimensional point matches PM with $PM_{ij} \subset \mathbb{R}^n \times \mathbb{R}^n$ and a set of fixed views $F \subseteq V$ by minimizing:

$$\underset{R \setminus \{R_i | V_i \in F\}}{\arg\min} \sum_{(i,j) \in C} \left(\sum_{(x_k, y_k) \in PM_{ij}} ||R_i(x_k) - R_j(y_k)||^2 \right) \tag{3}$$

Note that for all fixed views, the registration will be constrained to be the identity transformation I: $\forall V_i \in F : R_i = I$.

Global optimization given pairwise transformations

The intensity-based pairwise shift calculations do not directly give us the point correspondences we need for the global optimization step, instead the results are pairwise transformations T^p in the form of affine transform matrices. We can, however, easily construct point correspondences by taking a set of points and transforming them with the *inverse* transform (the only requirement being that the n-dimensional points do not all lie in a subspace of lower dimensionality of $\mathbb{R}^n$).

Using the 3-dimensional pairwise transformations T^p ($T^p_{ij}(x_j) = x_i$) between two image (groups) V_i and V_j given their existing registrations R^{meta}, we use the 8-point approximate bounding box of their overlapping region BB_{ij} to construct the point correspondences: $PM_{ij} = \{\big(bb_k, (T^p_{ij})^{-1}(bb_k)\big) | bb_k \in BB_{ij}\}$. We can then determine the globally optimal registrations R by performing the minimization described above (3).

Global optimization with iterative link dropping

Once the global optimization terminates due to convergence or exceeding of the maximum number of iterations, we can calculate the *error* of the individual images as the average displacement of all interest points in an image to their point matches:

$$e_i = \frac{\sum_{\{j:(i,j) \in C\}} \sum_{(x_k, y_k) \in PM_{ij}} ||R_i(x_k) - R_j(y_k)||}{\sum_{\{j:(i,j) \in C\}} |PM_{ij}|} \tag{4}$$

If the link graph (V, C^n) contains links with contradicting point correspondences, stopping after one round of global optimization might leave us with unsatisfying results. In the *iterative* version of the global optimization, we therefore check that both the average error of all images and the ratio of maximal and average error fall below a user-defined threshold. If these conditions are not yet met, we will proceed to iteratively remove disagreeing links from the link graph and repeat the global optimization. To do this, we first determine the link with the highest error by maximizing:

$$c_{worst} = \underset{(i,j)}{\arg\max} \max_{(x_k, y_k) \in PM_{ij}} \left((1 - q_{ij})^2 \sqrt{d_{ijk}} \log_{10} \Big(\max \big(deg(i), \deg(j) \big) \Big) \right) \tag{5}$$

with d_{ijk} denoting the distance of the k'th point match of the link (i,j), $d_{ijk} = ||R_i(x_k) - R_j(y_k)||$, $\deg(i)$ denoting the degree (number of neighbors) of an image V_i in the link graph and q_{ij} being a *quality metric* $\in (0,1)$ of the link, e.g. 0-truncated cross correlation. We then remove the worst link from the links ($C^{n+1} \leftarrow C^n \setminus c_{worst}$) and repeat the optimization step 3 with the new link graph (V, C^{n+1}). The whole process is repeated until the errors fall below a user-defined threshold (in the worst case, links will be dropped until we end up with *spanning trees* of the connected components in the link graph).

Two-round global optimization using metadata

If some cases, the link graph might contain multiple connected components, e.g. in datasets from screening applications, where the actual sample only occupies isolated ”islands” and most images contain only background. In this case, we can only reliably determine pairwise transformations within the connected components and align images within the components in the global optimization step. We might, however, have reasonable registrations R^{meta} from metadata and wish to keep as closely as possible to those if we do not have *strong* links.

For this, we developed a *two-round* version of the global optimization. In the first round, we determine registrations R^{strong} as described above, using the graph of *strong* links, i.e. links that are backed by pairwise transformations. In the second round, we determine the connected components in the (V, C^{strong}) graph and a mapping $CC : \mathbb{N} \rightarrow \mathbb{N}$ from image (group) indices to connected component indices as well as *weak links* $C^{weak} = \{(i,j) \in V \times V | CC(i) \neq CC(j)\}$ between images in different components. We then determine transformations R^{cc} for each connected component not containing a fixed image by minimizing:

$$\underset{R^{cc} \setminus \{r_i^{cc} \in R^{cc} | CC_i \cap F \neq \emptyset\}}{\arg\min} \sum_{(i,j) \in C^{weak}} \sum_{bb_k \in BB_{ij}} ||R^{cc}_{CC(i)}\big(R_i^{strong}(bb_k)\big) - R^{cc}_{CC(j)}\big(R_j^{strong}(bb_k)\big)||^2 \tag{6}$$

Note that we use the corners bb_k of the bounding box BB_{ij} of the overlapping volume of two images V_i and V_j as the point correspondences. The overlap is determined according to the metadata transformations R^{meta} and we essentially try to ”un-do” the registrations of the first round as well as possible (while keeping the registrations *within* the connected components). The final transformations R are the concatenation of the registrations within the connected components with the relative transformations of the connected components: $R_i \leftarrow R^{cc}_{CC(i)} R_i^{strong}$.

13. Simulation of light propagation in tissue using raytracing

To describe the scene we will simulate we use two phantom images of the same size that separately define the visible light image (corresponding to fluorescent probe distribution) and the refractive indices map (**Fig. 2e,f**). We deliberately embed the spheroid-like object of varying refractive index ($R_i = 1.1$–1.21) within a dense, invisible material with high refractive index ($R_i = 1.1$) surrounded by air ($R_i = 1.0$) to recapitulate significant aberrations in the illumination light path using a relatively small simulation volume of 289×289×289px. The object simulations are implemented in the multiview-simulations package.[13]

We virtually scan a concave lightsheet (diameter of 1 pixel in the center, and 3 pixels at the edge) in 1-pixel steps and alternating left and right illumination through the sample (**Fig. 2g**), simulating an entire volume for each lightsheet position and direction (**Fig. 2h** and **Supplementary Video 2**). Therefore, we send 200.000 individual rays originating from random positions within the concave lightsheet through the sample for each lightsheet simulation. The initial vector of each ray points approximately along the lightsheet illumination direction and moves in 1-pixel steps through the volume. After each move we locally compute the Eigenvector of the largest Eigenvalue using the refractive index map, which defines the normal vector of the refractive surface at the current, sub-pixel ray position. Using this estimated refraction surface, we compute the refraction angle using raytracing algebra,[20] update the ray vector accordingly, and add a Gaussian distribution with an intensity defined by the visible light image to the simulation volume. For simplicity we ignore total reflection since it is mostly caused by numerical instabilities. We confirmed correct refraction of rays based on our computation of local Eigenvectors in discrete pixel-images by comparing it to refraction of rays in the corresponding continuous, parametric description of the same scene (not shown).

The result of these simulations are 578 3d-volumes that recapitulate the principles of dual-sided lightsheet illumination (**Fig. 2g**). Inspired by classical raytracing, we perform a simplified detection simulation and therefore invert the ray path and only modulate signal intensity as a function of distance from the focal plane. Per camera pixel (289×289) we send 500 rays at random positions within each pixel into the scene that are refracted as described above. For detection, we use the same the same refractive index map, and the result of each respective lightsheet illumination simulation serves as image data. However, since we assume an extremely high refractive index mismatch for illumination simulation to recapitulate the behavior in large samples, we assume a lower refractive index mismatch for the embedding material ($R_i = 1.01$) to acquire reasonably distorted images. The relative refractive index mismatch within the spheroid-like object is conserved ($R_i = 1.01$–1.11). We assume the focal point of the objective to lie in the center of the currently simulated lightsheet position. The light captured by each ray on its path through the sample is then computed as the sum of all light integrated when traveling through the sample, at each ray location gaussian-weighted ($\sigma = 3.5$) by the distance to the expected lightsheet position. The simulations were performed in parallel on the local compute cluster at the MDC.

Simulated data was created using the $net.preibisch.simulation.SimulateMultiViewAberrations$ class in the multiview-simulation package (release version 0.2.2). Since it is a Maven artifact, the versions of all dependencies are defined and the corresponding version can be built automatically from that source code state (`https://github.com/PreibischLab/multiview-simulation/commit/b41b74cce9287f804b670d7de3396605446818a8`).

14. Non-rigid transformation

The underlying principle of moving least squares[21] is to non-rigidly transform images using a set of corresponding points. Therefore, a local transformation is computed for each pixel using a distance-weighted fit of all corresponding points ensuring smoothness. In BigStitcher, corresponding points are a direct result of all interest point-based registration algorithms. To provide a sufficient amount of corresponding interest points, it is yet most useful to derive them using ICP.[14] Regularization is achieved on the registration side as corresponding interest points are identified on a regularized affine transformation model either using RANSAC[12] or ICP,[14] which both specify a maximum error. This ensures that corresponding points cannot diverge more than this specified error from the regularized affine transformation of each image tile. In combination with virtual re-blocking, this error can be limited to smaller regions than the acquired, physical tiles.

When computing local transformations for each image, it is necessary to ensure smoothness across n overlapping images by defining appropriate point correspondences. However, the registration identifies only pairwise correspondences in between pairs of images. From those, we therefore first identify all unique interest points across all images defined by all pairwise correspondences (**Supplementary Fig. 23**). The location of each unique point is then determined by averaging the locations of all contributing interest points after applying their respective affine transformations. Thereby, the non-rigid transformation only accounts for the remaining error after affine alignment. For each image, corresponding points required for moving least squares are then subsequently defined between the unique point and the corresponding interest point of the transformed image only.

15. Quality estimation using relative Fourier Ring Correlation

For computing the 2d-Fourier Ring Correlation[22] we adapted methods from the FRC ImageJ plugin[23] as outlined in the **Online Methods**.

16. Image fusion

We *fuse* multiple images by performing a weighted average of the raw images I^{raw} transformed by their registrations R. Each raw image I_i^{raw} has a set weight images W_i. For example, we allow the user to weigh the images with a cosine-shaped fade-out, de-emphasizing the artifact-prone border regions of the individual images, as well as by the approximate local entropy, to emphasize images with sharper structures in overlapping regions. Since the raw images will be evaluated at non-integer coordinates, we offer the choice between nearest-neighbor and linear interpolation. Downsampling can easily be achieved by prepending a scaling transformation to each of the registrations R. The intensity of the fused volume at a coordinate x is given by:

$$I^{fused}(x) = \frac{\sum_{I_i^{raw} \in I^{raw}} \left(I_i^{raw}\left(R_i^{-1}(x)\right) * \prod_{w_j \in W_i} w_j\left(R_i^{-1}(x)\right)\right)}{\sum_{I_i^{raw} \in I^{raw}} \left(\prod_{w_j \in W_i} w_j\left(R_i^{-1}(x)\right)\right)} \tag{7}$$

In practice, we evaluate I^{fused} only at integer coordinates of a user-defined *bounding box*. We implemented the image fusion to perform all calculations virtually on-the-fly, with caching of previously computed planes using imglib2-cache. This allows the quick inspection of fusion results as well as creation and planewise saving of images that might exceed the RAM available to the user.

17. Brightness and contrast adjustment

Even after correcting for fixed-pattern noise (5), differences in brightness and contrast between images, e.g. due to bleaching, might persist and be visible in the fused images. To correct for this, we estimate optimal linear transforms of pixel intensities in adjacent images[24] to achieve uniform brightness and contrast in the whole dataset. We minimize the intensity difference of all pixels in the overlapping volume O_{AB} of two images I_A, I_B (with corresponding coordinates (x_A, x_B) according to the current registrations):

$$\underset{\alpha,\beta}{\arg\min} \sum_{I_A \in I} \left(\sum_{I_B \in I \setminus I_A} \left(\sum_{(x_A,x_B) \in O_{AB}} \left(I_B(x_B) - \left[\alpha^{I_A} I_A(x_A) + \beta^{I_A}\right]\right)^2 \right) \right) \tag{8}$$

Since this is equal to one-dimensional point correspondence registration, we can make use of the same iterative optimization algorithm used for image registration (12). To reduce influence of noise and computational costs, we use (precomputed) downsampled versions of the images for the optimization. A problem with unconstrained optimization is the possibility of convergence to the trivial solution of setting all pixel intensities to zero. We therefore formulate the linear transform $I(x) * \alpha + \beta$ as a weighted average between a linear transform, an additive transform and the identity transform:

$$\alpha I(x) + \beta = \lambda_1 * (\alpha' I(x) + \beta_1) + \lambda_2 * (I(x) + \beta_2) + \lambda_3 * I(x) \tag{9}$$

with user-definable regularization parameters $\lambda_1, \lambda_2, \lambda_3 : \lambda_1 + \lambda_2 + \lambda_3 = 1$. By using nonzero λ_2, λ_3, we can constrain the optimization to not converge to the trivial solution.

The size of overlaps between image tiles can differ significantly. Therefore, intensity transformations supported by many overlapping pixels will implicitly have a higher weight, which can lead to the fact that visible intensity differences between tiles with little overlap persist. To compensate this effect we allow to balance overlaps by setting a maximal number of corresponding pixels. To ensure equal distribution of these corresponding pixels, we randomly remove pixels from the set of all pixels until the desired number is achieved.

18. Example datasets for BigStitcher

We prepared three different examples of different size and complexity for testing the BigStitcher. We suggest to run BigStitcher on these first before applying it to your dataset. This allows you to quickly test features in an environment where you can easily ask for advice on GitHub or the ImageJ Forum.

The data can be downloaded from the Open Science Foundation (a Nature recommended data repository https://www.nature.com/sdata/policies/repositories) at https://osf.io/bufza/.

2d multi-tile dataset (2.8 MB)

Maximum intensity projection of the nervous system of a Drosophila larva containing 6 tiles and 3 channels each. You can download the raw input at http://preibischlab.mdc-berlin.de/BigStitcher/Grid_2d.zip and a reconstructed BigStitcher project at http://preibischlab.mdc-berlin.de/BigStitcher/Grid_2d_h5_aligned.zip. In the reconstructed project, the images were imported into the BigStitcher using the AutoLoader (with immediate resaving as HDF5 and Movement to a regular 2-by-3 grid with 10% overlap between the tiles). We calculated pairwise shifts using phase correlation with default parameters, using the precomputed 2x2 downsampling and averaging the channels. We ignored links with correlation < 0.7 and calculated the final registration using the two-round global optimization with strict constraints.

3d multi-tile dataset (123 MB)

3d confocal scan of the nervous system of a Drosophila larva containing 6 tiles and 3 channels each, channels are distributed over different files. You can download the raw input at http://preibischlab.mdc-berlin.de/BigStitcher/Grid_3d.zip and the reconstructed project at http://preibischlab.mdc-berlin.de/BigStitcher/Grid_3d_h5_aligned.zip. In the reconstructed project, we ran the same import and reconstruction steps as for the 2d dataset and in addition performed affine refinement of the registration using IPC with default parameters and simple tile refinement to create the final reconstructed project.

We will add larger and more complex examples on the BigStitcher website https://imagej.net/BigStitcher and will also link videos of the alignment process from there.

19. Links to the current source codes

The BigStitcher is distributed over two projects. Both are licensed under the GPL(v2) and the source code is freely available on GitHub, at `https://github.com/PreibischLab/BigStitcher` and `https://github.com/PreibischLab/multiview-reconstruction`, respectively.

The CUDA code for accelerated interest point detection and devonvolution is available from `https://github.com/StephanPreibisch/SeparableConvolutionCUDALib` and `https://github.com/StephanPreibisch/FourierConvolutionCUDALib`, respectively.

The light simulation is a standalone software and part of the multiview-simulation package `https://github.com/PreibischLab/multiview-simulation`, the main class can be found here: `https://github.com/PreibischLab/multiview-simulation/blob/master/src/main/java/net/preibisch/simulation/SimulateMultiViewAberrations.java`. The license is also GPL(v2).

Newer versions will be hosted using GitHub, and release announcements will be done via Twitter (`https://twitter.com/preibischs`), on the GitHub page (`https://github.com/PreibischLab/BigStitcher`), and on the ImageJ wiki (`http://imagej.net/BigStitcher`). Releases are and will be provided to end users via the Fiji update mechanism.[25]

The following classes are the main classes for the respective codes:

BigStitcher: $net.preibisch.stitcher.plugin.BigStitcher$ (in BigStitcher)

Light simulation: $net.preibisch.simulation.SimulateMultiViewAberrations$ (in multiview-simulation)

Phase correlation simulation: $net.preibisch.stitcher.headless.StitchingPairwise$ (in BigStitcher)

20. Bug reports and feature requests

For bug reports and feature requests regarding BigStitcher please use the GitHub issue system available here: `https://github.com/preibischLab/BigStitcher/issues`.

21. BigStitcher user guide

The BigStitcher comes with extensive documentation that is hosted on the ImageJ wiki. The current version of the continuously updated user guide can be found at `https://imagej.net/BigStitcher#Documentation`.

REFERENCES

[1] Chung, K., Wallace, J., Kim, S.-Y., Kalyanasundaram, S., Andalman, A. S., Davidson, T. J., Mirzabekov, J. J., Zalocusky, K. A., Mattis, J., Denisin, A. K., Pak, S., Bernstein, H., Ramakrishnan, C., Grosenick, L., Gradinaru, V., and Deisseroth, K., "Structural and molecular interrogation of intact biological systems," *Nature* **497**(7449), 332–337 (2013).

[2] Chhetri, R. K., Amat, F., Wan, Y., Höckendorf, B., Lemon, W. C., and Keller, P. J., "Whole-animal functional and developmental imaging with isotropic spatial resolution," *Nature Methods* **12**, 1171–1178 (Oct. 2015).

[3] Linkert, M., Rueden, C. T., Allan, C., Burel, J.-M., Moore, W., Patterson, A., Loranger, B., Moore, J., Neves, C., MacDonald, D., et al., "Metadata matters: access to image data in the real world," *The Journal of Cell Biology* **189**(5), 777–782 (2010).

[4] Pietzsch, T., Preibisch, S., Tomančák, P., and Saalfeld, S., "ImgLib2—generic image processing in Java," *Bioinformatics* **28**(22), 3009–3011 (2012).

[5] Pietzsch, T., Saalfeld, S., Preibisch, S., and Tomancak, P., "BigDataViewer: visualization and processing for large image data sets," *Nature methods* **12**(6), 481 (2015).

[6] Preibisch, S., Saalfeld, S., and Tomancak, P., "Globally optimal stitching of tiled 3D microscopic image acquisitions," *Bioinformatics* **25**(11), 1463–1465 (2009).

[7] Kuglin, C., "The phase correlation image alignment method," *IEEE International Conference on Cybernetics and Society* (1975).

[8] Lowe, D. G., "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision* **60**(2), 91–110 (2004).

[9] Baker, S. and Matthews, I., "Lucas-Kanade 20 years on: A unifying framework," *International Journal of Computer Vision* **56**(3), 221–255 (2004).

[10] Preibisch, S., Saalfeld, S., Schindelin, J., and Tomancak, P., "Software for bead-based registration of selective plane illumination microscopy data," *Nature Methods* **7**(6), 418 (2010).

[11] Smith, C. S., Preibisch, S., Joseph, A., Abrahamsson, S., Rieger, B., Myers, E., Singer, R. H., and Grunwald, D., "Nuclear accessibility of β-actin mRNA is measured by 3D single-molecule real-time tracking," *J Cell Biol* **209**(4), 609–619 (2015).

[12] Fischler, M. A. and Bolles, R. C., "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM* **24**, 381–395 (June 1981).

[13] Preibisch, S., Amat, F., Stamataki, E., Sarov, M., Singer, R. H., Myers, E., and Tomancak, P., "Efficient Bayesian-based multiview deconvolution," *Nature Methods* **11**(6), 645 (2014).

[14] Besl, P. J. and McKay, N. D., "A method for registration of 3-d shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **14**, 239–256 (Feb 1992).

[15] Richardson, W. H., "Bayesian-based iterative method of image restoration," *JOSA* **62**(1), 55–59 (1972).

[16] Lucy, L. B., "An iterative technique for the rectification of observed distributions," *The astronomical journal* **79**, 745 (1974).

[17] Tikhonov, A. N. and Arsenin, V. Y., [*Solutions of Ill-posed Problems*], W.H. Winston (1977).

[18] Saalfeld, S., Cardona, A., Hartenstein, V., and Tomančák, P., "As-rigid-as-possible mosaicking and serial section registration of large ssTEM datasets," *Bioinformatics* **26**(12), i57–i63 (2010).

[19] Saalfeld, S., Fetter, R., Cardona, A., and Tomancak, P., "Elastic volume reconstruction from series of ultra-thin microscopy sections," *Nature Methods* **9**(7), 717 (2012).

[20] De Greve, B., "Reflections and refractions in ray tracing." `https://graphics.stanford.edu/courses/cs148-10-summer/docs/2006--degreve--reflection_refraction.pdf` (2006). Accessed: 2018-11-30.

[21] Schaefer, S., McPhail, T., and Warren, J., "Image deformation using moving least squares," *ACM Trans. Graph.* **25**, 533–540 (July 2006).

[22] Nieuwenhuizen, R. P. J., Lidke, K. A., Bates, M., Puig, D. L., Grunwald, D., Stallinga, S., and Rieger, B., "Measuring image resolution in optical nanoscopy," *Nature Methods* **10**, 557–562 (Apr. 2013).

[23] Burri, O. and Herbert, A., "Fourier ring correlation plugin." `https://imagej.net/Fourier_Ring_Correlation_Plugin` (2018). Accessed: 2018-11-08.

[24] Blasse, C., Saalfeld, S., Etournay, R., Sagner, A., Eaton, S., and Myers, E. W., "PreMosa: extracting 2D surfaces from 3D microscopy mosaics," *Bioinformatics* **33**(16), 2563–2569 (2017).

[25] Schindelin, J., Arganda-Carreras, I., Frise, E., Kaynig, V., Longair, M., Pietzsch, T., Preibisch, S., Rueden, C., Saalfeld, S., Schmid, B., et al., "Fiji: an open-source platform for biological-image analysis," *Nature Methods* **9**(7), 676 (2012).