Developmental Cell, Volume *36*

## Supplemental Information

# Real-Time Three-Dimensional Cell Segmentation in Large-Scale Microscopy Data of Developing Embryos

**Johannes Stegmaier, Fernando Amat, William C. Lemon, Katie McDole, Yinan Wan, George Teodoro, Ralf Mikut, and Philipp J. Keller**

## *Supplemental Inventory*

### Supplemental Figures

| | |
|---|---|
| **Figure S1**, related to Figure 1 | RACE segmentation workflow and parameter sensitivity analysis |
| **Figure S2**, related to Figure 1 | Fast and accurate slice-based 2D segmentation |
| **Figure S3**, related to Figure 1 | Efficient seed-based fusion of 2D segments to 3D cell shapes |
| **Figure S4**, related to Figure 1 | Comparative analysis of precision, recall, processing time and scalability |
| **Figure S5**, related to Figure 5 | Accuracy of cell shape information extracted from confocal *Drosophila* images |
| **Figure S6**, related to Figure 6 | Comparison of *Drosophila* whole-embryo tissue anisotropy maps |
| **Figure S7**, related to Figure 7 | Segmentation quality in gastrulating *Drosophila* wild-type and *bnt* mutant embryos |

### Supplemental Tables

| | |
|---|---|
| **Table S1**, related to Figure 1 | Comparison of processing time for CPU-optimized and GPU-accelerated implementations of RACE |
| **Table S2**, related to Figure 1 | Parameter settings used in the segmentation quality comparison |
| **Table S3**, related to Figure 1 | Segmentation quality comparison and analysis of cell shape information accuracy |

### Supplemental Movies

| | |
|---|---|
| **Movie S1**, related to Figure 2 | Detection of seed points in *Drosophila* |
| **Movie S2**, related to Figure 3 | Cell segmentation in *Drosophila*, zebrafish and mouse embryos |
| **Movie S3**, related to Figure 7 | Tissue anisotropy mapped at the single-cell level in *Drosophila* wild-type and *bnt* mutant embryos |
| **Movie S4**, related to Figure 7 | Side-by-side comparison of tissue anisotropy in *Drosophila* wild-type and *bnt* mutant embryos |
| **Movie S5**, related to Figure 8 | Joint reconstruction of cell lineages and cell morphology in early *Drosophila* development |

### Supplemental Software

| | |
|---|---|
| **Software S1**, related to Figure 1 | RACE cell segmentation framework for Windows, Mac OS X and Ubuntu |

### Supplemental Experimental Procedures

### Supplemental References

**Figure S1 |** RACE segmentation workflow and parameter sensitivity analysis

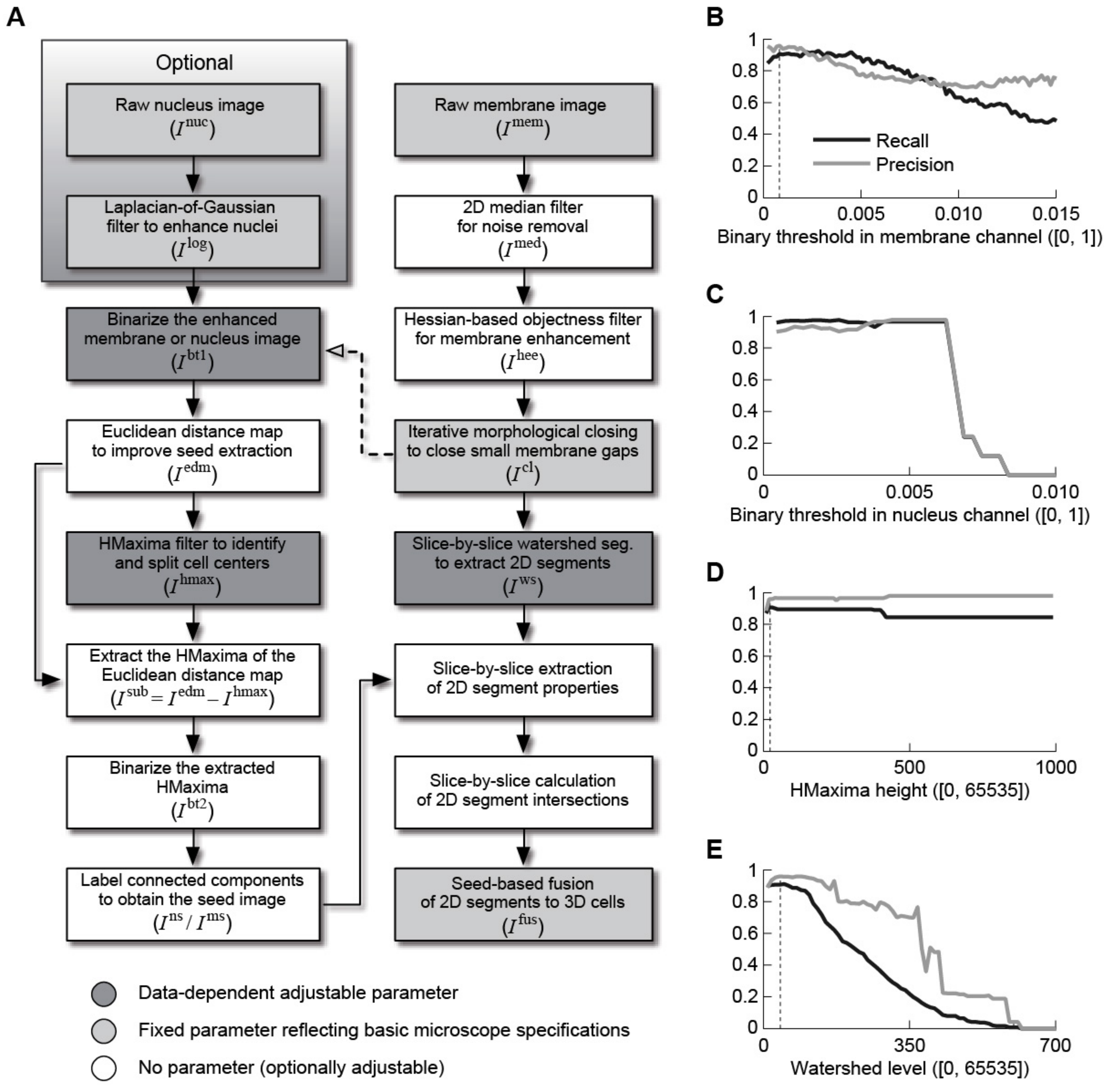

A
Optional
Raw nucleus image ($I^{nuc}$)
Laplacian-of-Gaussian filter to enhance nuclei ($I^{log}$)
Binarize the enhanced membrane or nucleus image ($I^{bt1}$)
Euclidean distance map to improve seed extraction ($I^{edm}$)
HMaxima filter to identify and split cell centers ($I^{hmax}$)
Extract the HMaxima of the Euclidean distance map ($I^{sub} = I^{edm} - I^{hmax}$)
Binarize the extracted HMaxima ($I^{bt2}$)
Label connected components to obtain the seed image ($I^{ns} / I^{ms}$)
Raw membrane image ($I^{mem}$)
2D median filter for noise removal ($I^{med}$)
Hessian-based objectness filter for membrane enhancement ($I^{hee}$)
Iterative morphological closing to close small membrane gaps ($I^{cl}$)
Slice-by-slice watershed seg. to extract 2D segments ($I^{ws}$)
Slice-by-slice extraction of 2D segment properties
Slice-by-slice calculation of 2D segment intersections
Seed-based fusion of 2D segments to 3D cells ($I^{fus}$)
Data-dependent adjustable parameter
Fixed parameter reflecting basic microscope specifications
No parameter (optionally adjustable)
B
Recall
Precision
Binary threshold in membrane channel ([0, 1])
C
Binary threshold in nucleus channel ([0, 1])
D
HMaxima height ([0, 65535])
E
Watershed level ([0, 65535])

**Figure S1**, related to Figure 1 | RACE segmentation workflow and parameter sensitivity analysis

(**A**) The left column of operators shows the processing steps for seed detection. The right column shows the core processing steps for cell segmentation, which rely on the detected seed segments. An optional pathway based on nuclear image data (if available) is indicated by the shaded box. If no cell nuclei data is available, the dashed pathway (pointing to the enhanced membrane image) indicates the workflow for generating seed segments. The hollow arrowhead indicates an image inversion step and image data referenced in parentheses are described in more detail in Part 1 of **Supplemental Experimental Procedures**. Only the three parameters of the operators shown in dark gray shading are data-dependent and need to be adjusted for a new data set. Parameters of processing steps shown with light gray shading reflect basic settings related to microscope or experiment configuration (such as pixel size in the image data, etc.). All other pipeline components use default parameter settings, which were kept constant throughout this study. However, these parameters can optionally also be adjusted.
(**B-E**) Sensitivity analysis of the core parameters of the RACE segmentation framework, using manual ground truth annotations in a *Drosophila* SiMView image data set (see also **Table S2**). The respective full valid parameter ranges are provided in square brackets. Panels (**B**), (**D**) and (**E**) were calculated using RACE (MS), *i.e.* using a membrane channel only and with heuristics disabled. Panel (**C**) was calculated using RACE (NS), *i.e.* using both membrane and nuclear channels and with heuristics disabled. For each of the analyses, one parameter was varied while the other parameters were set to the values listed in **Table S2**. Dashed lines indicate the respective optimal parameter settings used in Tab 1 of **Table S3**. The large plateau region visible in (**C**) enables the use of an adaptive threshold instead of a fixed threshold value. Note that seed extraction is performed using either membrane or cell nuclei image data, *i.e.* only one of the binary thresholds investigated in panels (**B**) and (**C**) is needed when executing the RACE algorithm.

**Figure S2 |** Fast and accurate slice-based 2D segmentation

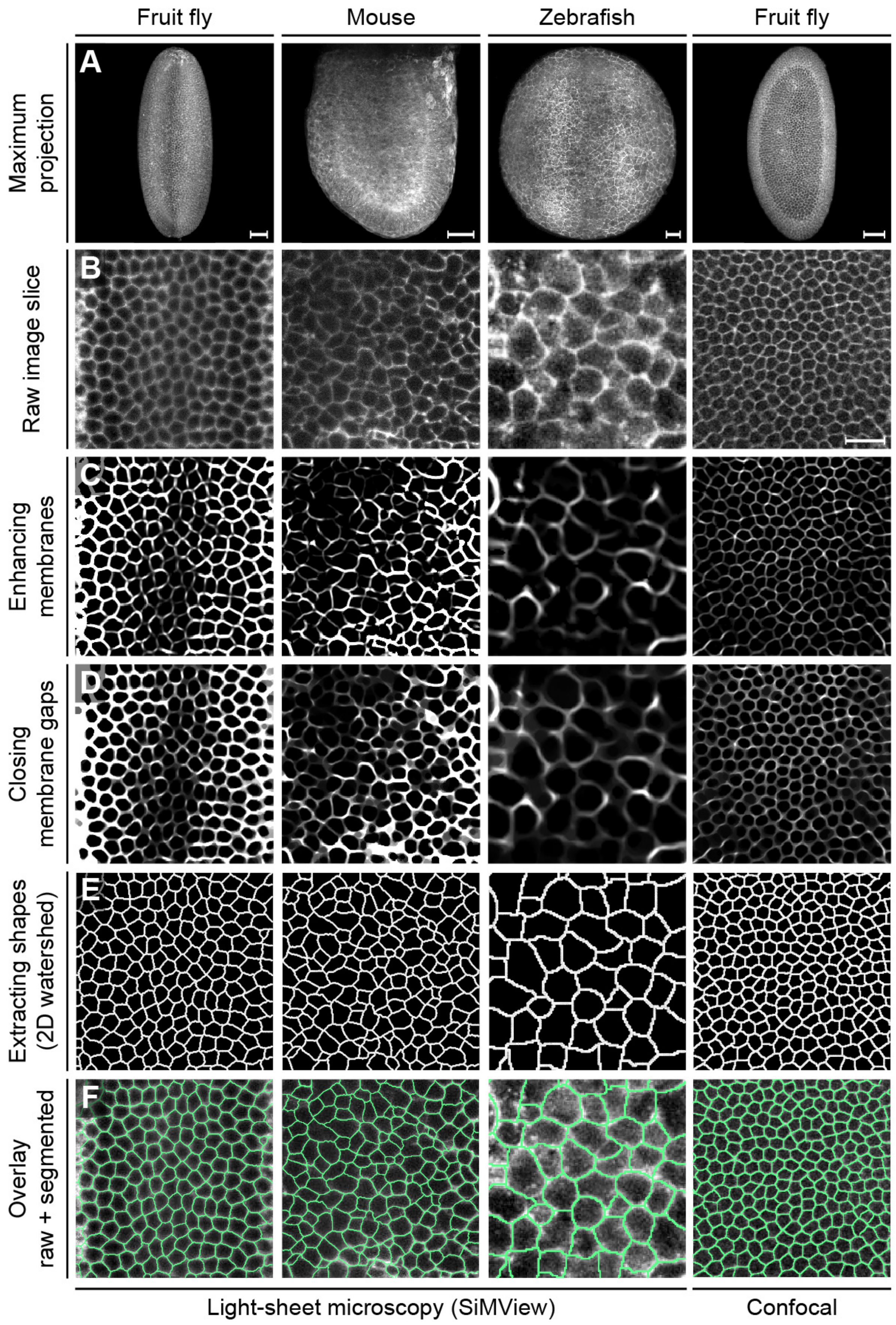


**Figure S2**, related to Figure 1 | Fast and accurate slice-based 2D segmentation

Illustration of the main processing steps performed prior to fusion of the 2D segments. Maximum intensity projections of 3D image stacks of *Drosophila*, mouse and zebrafish embryos recorded with SiMView light-sheet microscopy and confocal fluorescence microscopy are shown in panel (**A**). Based on the raw membrane image (**B**, xy-image is shown, i.e. an image slice oriented perpendicular to the microscope's optical detection axis), a Hessian-based objectness filter is used to enhance membrane structures (**C**). Remaining holes in the cell membranes are repaired using iterative morphological closing with increasing structure element sizes (**D**). The actual segmentation is performed using a standard watershed algorithm on the 2D slices (**E**). The last row (**F**) shows the watershed pixels superimposed with the raw images. Image contrast has been adjusted for better visibility.
Scale bars, 50 µm (**A**), 20 µm (**B**).

**Figure S3 |** Efficient seed-based fusion of 2D segments to 3D cell shapes

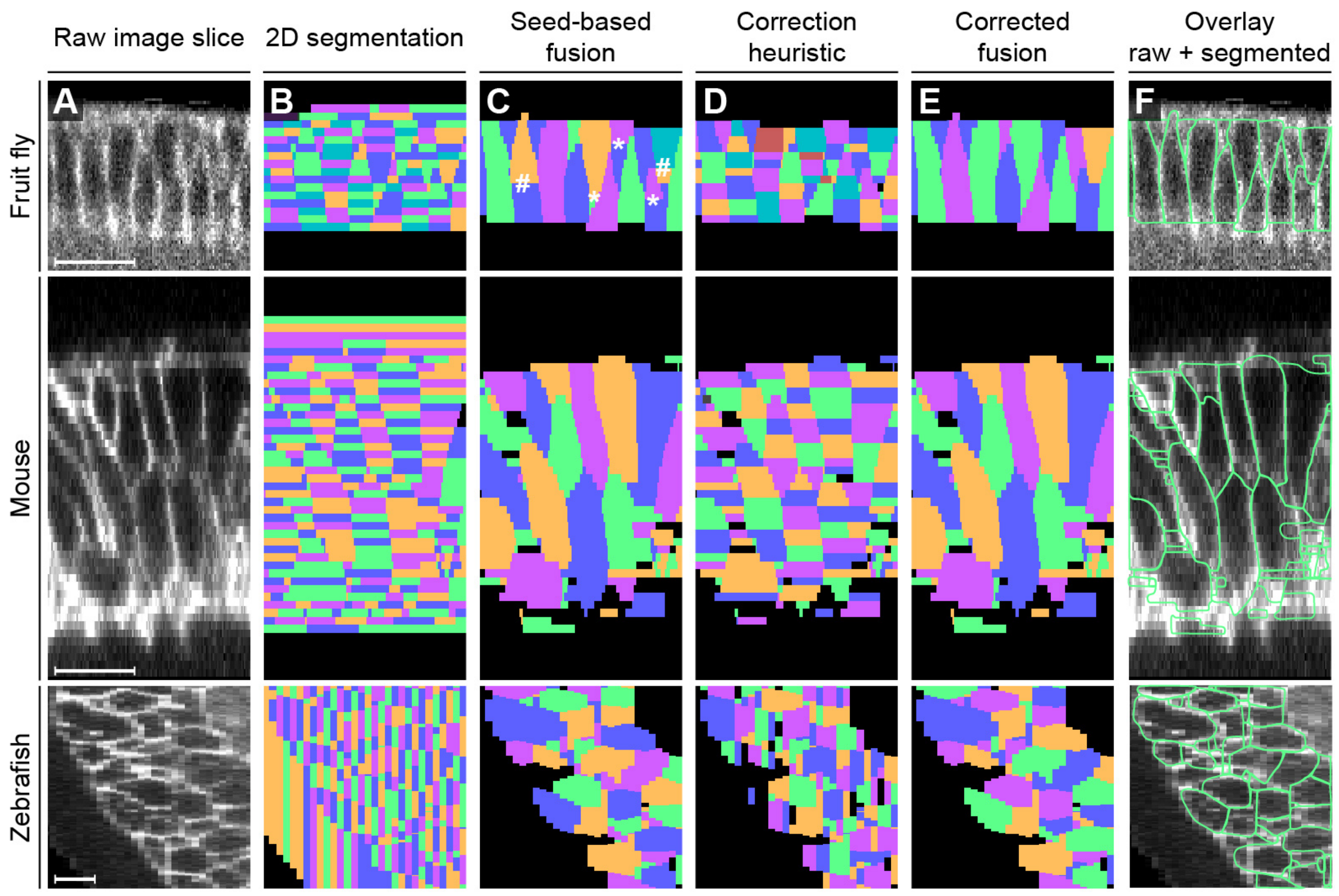
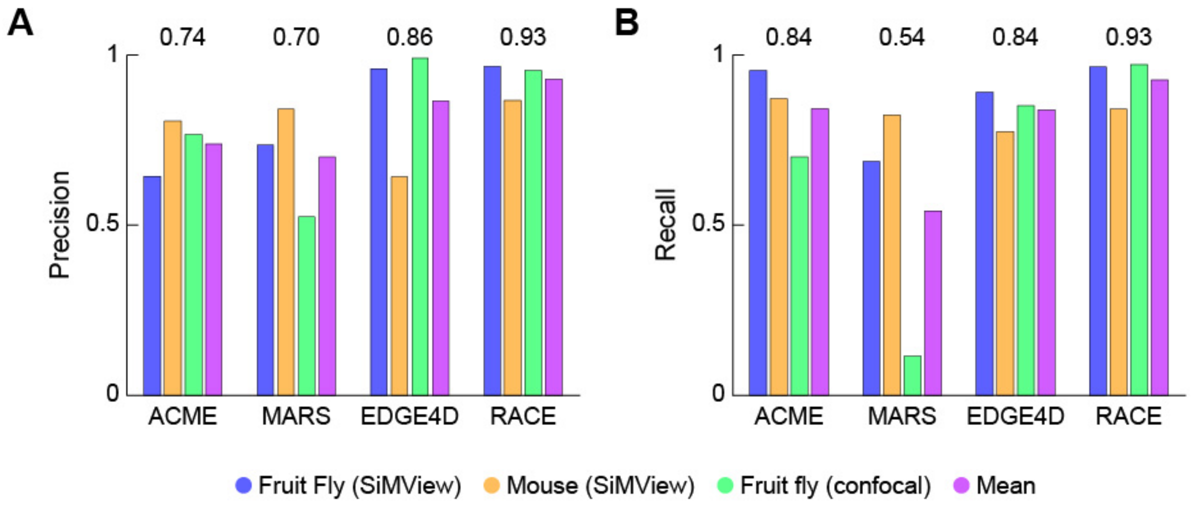


**Figure S3**, related to Figure 1 | Efficient seed-based fusion of 2D segments to 3D cell shapes

Illustration of final segment fusion steps for 3D cell shape segmentation in cell membrane image data (**A**, xz-image is shown, i.e. an image slice oriented parallel to the microscope's optical detection axis). The initial slice-based segmentation of the 2D morphological watershed operator (**B**) and the extracted seed segments are combined to reconstruct 3D cell shapes (**C**). 2D segments touching a seed segment are labeled with the unique seed identifier. Based on segment similarity across slices, 2D segments touching one of the labeled initialization segments are then iteratively merged to form complete 3D cell shapes, starting with the highest-scoring segments. In some cases, the 3D segmentation results (**C**) can be further improved by one of the two proposed fusion heuristics. Segments with few slices (indicated by * in panel (**C**)) are fused to their closest neighbor if the specimen-dependent maximum volume constraint is not violated by this merge. If the fusion of segments is suggested by the similarity-based fusion heuristic (**D**), two subcellular compartments produced in the initial seed-based segmentation (indicated by # in panel (**C**)) are fused to obtain the final 3D segmentation (**E**, **F**). Both fusion heuristics simply assign the same unique identifier to those segments that should be merged, thus producing a single corrected 3D cell shape. For specimens with diverse cell sizes, the fusion heuristics might not be able to further improve results (see e.g. the almost identical results obtained for the zebrafish and mouse panels shown before (**C**) and after (**E**) application of the fusion heuristics; see also Part 1 of **Supplemental Experimental Procedures**). We note that most of the small fragments visible in (**E**) and (**F**) are not over-segmentation artifacts but rather cell cross-sections belonging to cells whose centers are located outside the xz-slice shown here.

Scale bar, 20 μm.

**Figure S4 |** Comparative analysis of precision, recall, processing time and scalability



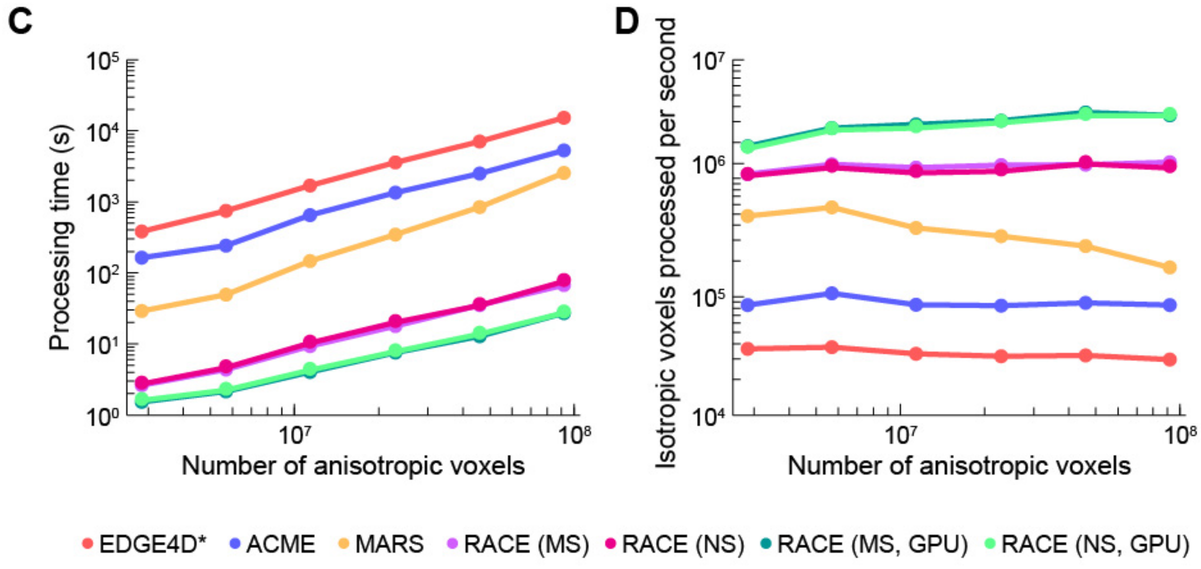


**Figure S4**, related to Figure 1 | Comparative analysis of precision, recall, processing time and scalability

(**A**, **B**) Comparison of segmentation quality obtained with ACME, MARS, EDGE4D and RACE. Segmentation quality was evaluated for a set of manually annotated images representing different model organisms (fruit fly embryo, mouse embryo) and microscopes (SiMView light-sheet microscopy, confocal fluorescence microscopy). RACE achieved comparable or superior segmentation accuracy in all investigated scenarios and provides highest average performance across model organisms and imaging modalities. Precision (**A**) and recall (**B**) values were derived from the topological errors made by the algorithms, where the sum of split and added cells was considered as the false positive count and the sum of merged and missing cells was considered as the false negative count (Part 2 of **Supplemental Experimental Procedures**). Average performance achieved across all model organisms and microscopes is indicated by the values shown above the respective group of bars representing each algorithm. This comparison is also summarized in **Figure 1B**, which shows average false discovery rates (1 – precision) and average false negative rates (1 – recall).
(**C**, **D**) Assessment of processing time (C) and voxel throughput (D) of ACME, MARS, EDGE4D, RACE (MS), RACE (NS), RACE (MS, GPU) and RACE (NS, GPU) for differently sized image regions taken from a Drosophila embryo image data set. The comparison includes image sizes 1316x628x111 (full embryo), 658x628x111 (half), 658x314x111 (1/4th), 658x314x55 (1/8th), 328x314x55 (1/16th) and 328x156x55 (1/32nd). ACME, MARS and EDGE4D generally perform up-sampling of the input image data to isotropic voxel size in order to meet algorithmic requirements. The results of this comparison show that all versions of RACE perform fastest by a large margin as they use discrete combinatorial optimization over 2D watershed regions, directly operate on the anisotropic image data and employ highly parallelizable processing steps. Processing time of all versions of RACE and most of the other investigated methods scales linearly with the number of voxels in the input data set. The comparison of processing times of the CPU-optimized and GPU-accelerated implementations of RACE shows that the substitution of CPU-based modules with high-performance GPU-accelerated modules for those processing steps representing the main computational bottlenecks further doubles the performance of our pipeline and enables real-time processing performance.
Performance was measured on a computer workstation equipped with two Intel Xeon E5 CPUs at 3.1 GHz (16 cores in total), 192 GB RAM, an NVidia Tesla K20 GPU and Windows 7 Professional 64-bit. The star (*) next to EDGE4D indicates that performance was measured on an Apple MacBook Pro equipped with an Intel Core i7 CPU at 2.3 GHz (4 cores in total), 16 GB memory and Mac OS X 10.9.5. We also tested EDGE4D on a high-end Mac Pro with two Intel Xeon E5 CPUs at 2.7 GHz but found that processing time did not change significantly, since very few operations are multi-threaded and time spent on I/O is minimal. Thus, EDGE4D processing time essentially depends only on the performance of a single CPU core.

**Figure S5 |** Accuracy of cell shape information extracted from confocal *Drosophila* images

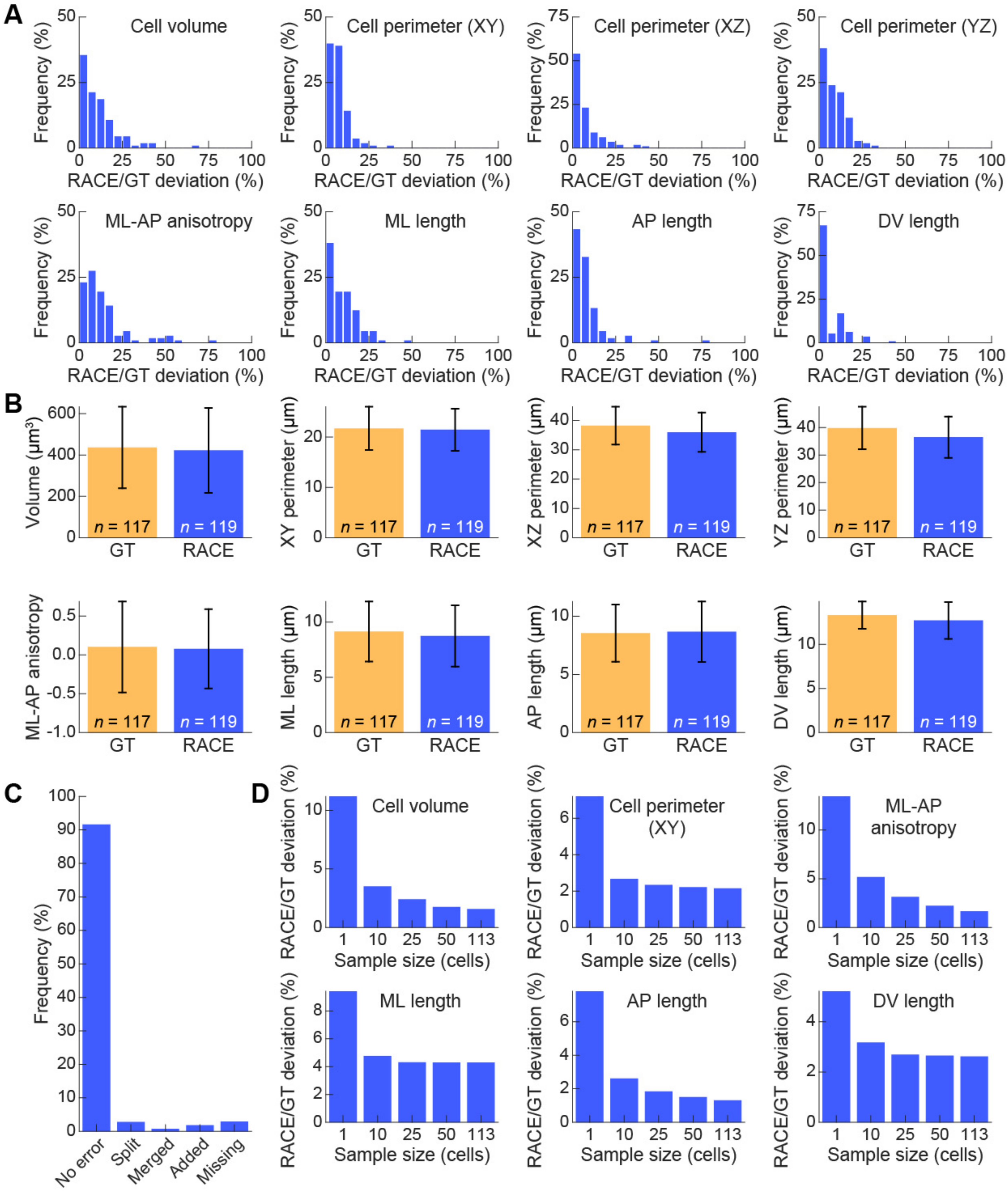


**Figure S5**, related to Figure 5 | Accuracy of cell shape information extracted from confocal *Drosophila* images

Similar to **Figure 5**, this figure shows an analysis of the accuracy of cell shape features automatically extracted by RACE, but for a *Drosophila* embryo imaged with a confocal fluorescence microscope. Please see the legend of **Figure 5** for a description of this analysis.

(**A**) Histogram of the absolute feature value deviation between automatic segmentation results and ground truth annotations, using a bin width of 5 %. The deviation of cell shape feature values is on average below 9 % (Tab 6 of **Table S3**).

(**B**) Bar plots of mean feature values measured across multiple regions of interest for both ground truth and automated RACE cell segmentation results. Error bars correspond to one standard deviation across the group of cells used for this analysis. We note that both average values and standard deviations are almost identical for RACE and GT annotations when combining data collected from small groups of cells.

(**C**) Bar plot of frequency of topological errors in RACE cell segmentation results. Topological errors are categorized as split, merged, added and missing cells.

(**D**) Deviation between RACE and GT annotations for six types of cell shape parameters, shown as a function of the sample size used for estimating average feature values. The special cases $n = 1$ and $n = 113$ show mean deviation of cell shape features at the single-cell level and averaged across all annotated cells, respectively. For settings in between these two special cases ($n = 10$, $n = 25$ and $n = 50$), 1,000 sub-groups were randomly selected from the data pool of all possible sub-groups containing matching cell pairs and deviation results were then averaged over all selected sub-groups. We note that measuring average cell shape features even for only a relatively modest number of cells already decreases the deviation between automated results and ground truth significantly. Additional features and results are presented in numerical form in Tab 6 of **Table S3**.

**Figure S6 |** Comparison of *Drosophila* whole-embryo tissue anisotropy maps

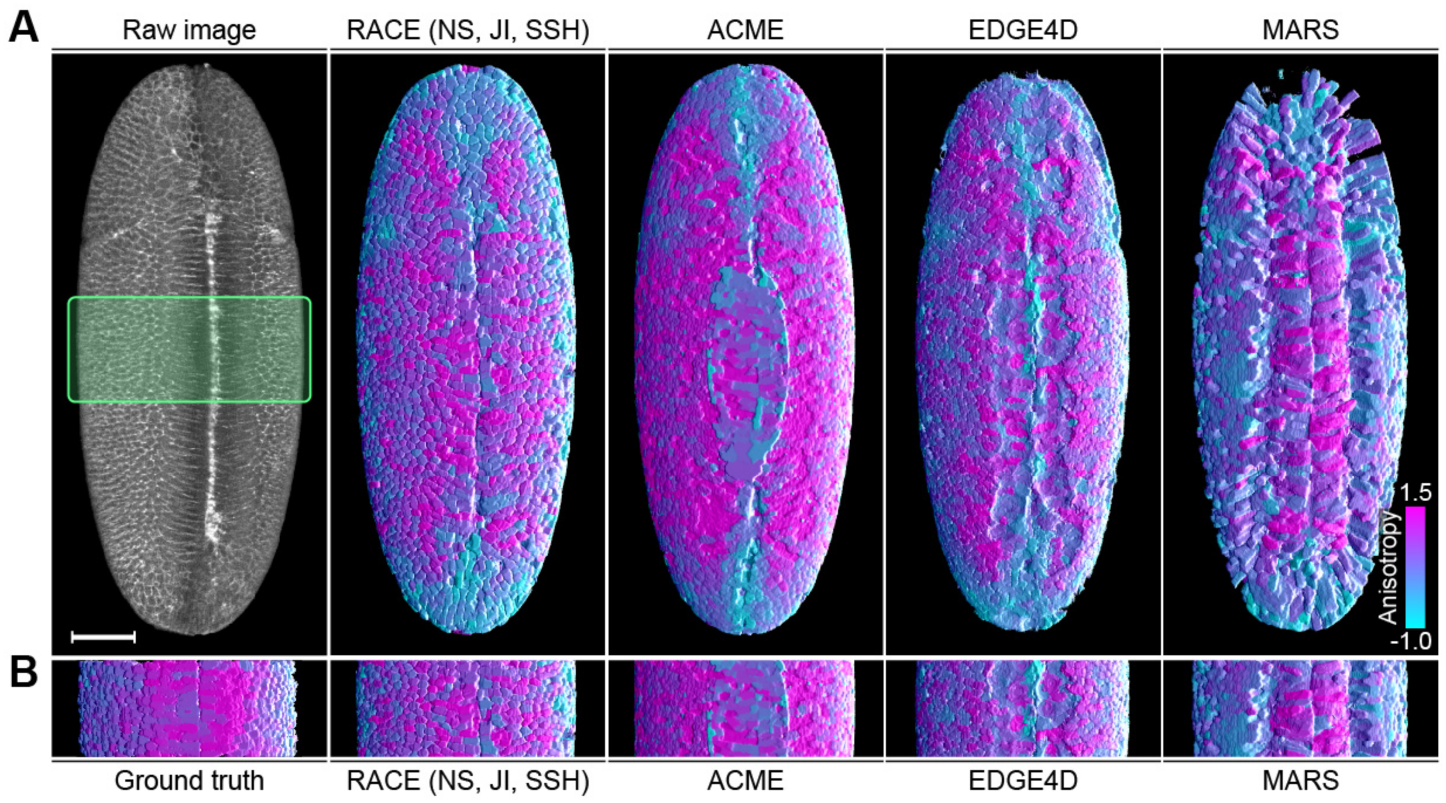


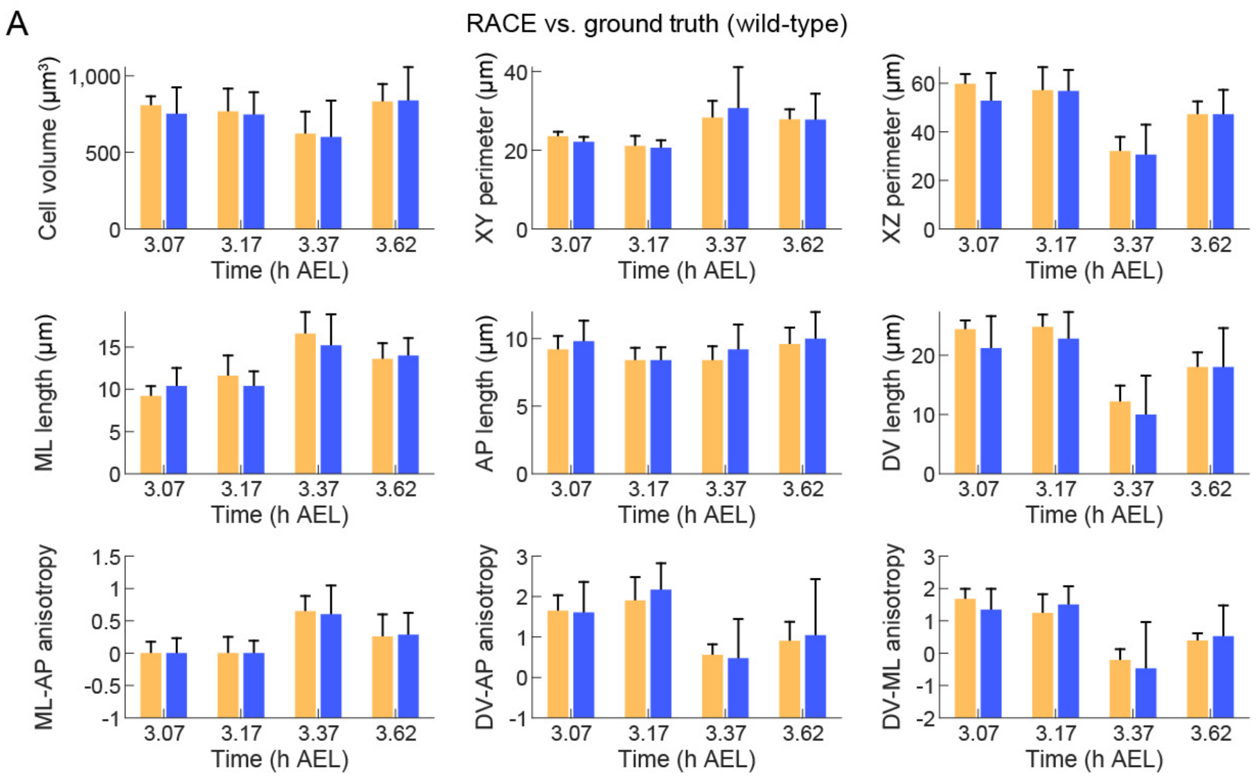
**Figure S6**, related to Figure 6 | Comparison of *Drosophila* whole-embryo tissue anisotropy maps

(**A**) Maximum-intensity projection of an image stack showing a membrane-labeled *Drosophila* embryo (left) and visualization of tissue anisotropy reconstructions across the *Drosophila* embryo (remaining panels) obtained with the segmentation methods RACE (NS, JI, SSH), ACME, EDGE4D and MARS, respectively. The 3D image stack was acquired at 3.25 hours after egg laying (AEL), using SiMView light-sheet microscopy. The calculation and color mapping of anisotropy levels was performed as described in Part 4 of **Supplemental Experimental Procedures**.

(**B**) Ground truth anisotropy map obtained from a manually segmented image region (first panel on the left side, corresponding to the region marked by the green rectangle in (**A**)) and the corresponding maps obtained by automatic image segmentation (remaining panels).

Scale bar, 50 µm.

**Figure S7 |** Segmentation quality in gastrulating *Drosophila* wild-type and bnt mutant embryos



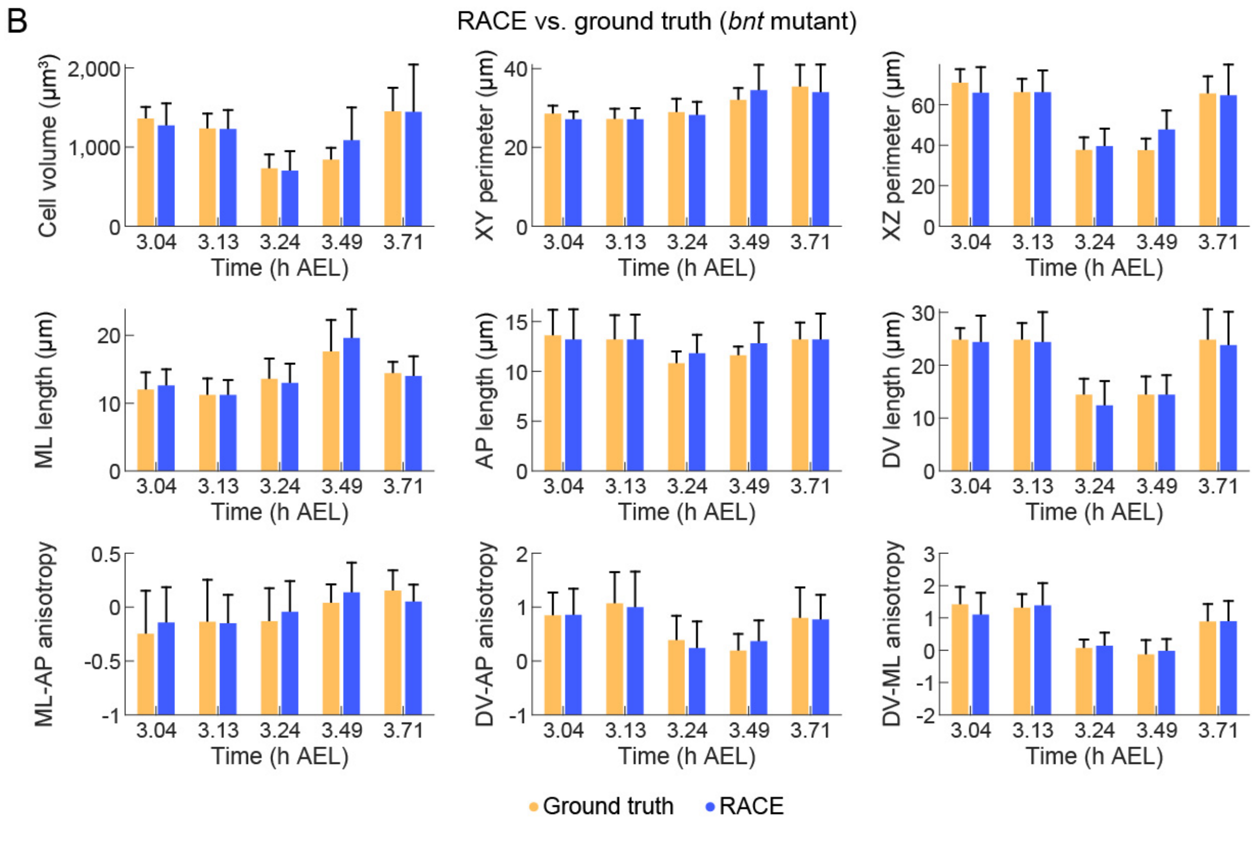


**Figure S7**, related to Figure 7 | Segmentation quality in gastrulating *Drosophila* wild-type and *bnt* mutant embryos

Evaluation of the accuracy of cell shape features automatically extracted by RACE, using segmentation data from *Drosophila* wild-type (**A**) and *bnt* mutant (**B**) embryos for the period 3.00-3.75 h AEL. At each time point a group of 20-30 cells was manually annotated and median values (colored bars) as well as standard deviation (error bars) of cell shape features across the group were extracted and visualized side-by-side with the respective results obtained from automatic reconstructions performed with RACE. Cell shape features investigated in this evaluation include cell volume (number of voxels), cell perimeter (number of surface voxels in xy-, xz- and yz-planes at the cell centroid location), mediolateral (ML) cell size, anteroposterior (AP) cell size and dorsoventral (DV) cell size. Moreover, mediolateral-vs.-anteroposterior cell shape anisotropy (ML-AP anisotropy), the mediolateral-vs.-dorsoventral cell shape anisotropy (ML-DV anisotropy) and dorsoventral-vs.-anteroposterior cell shape anisotropy (DV-AP anisotropy) were calculated as described in Part 4 of **Supplemental Experimental Procedures**. Median values as well as standard deviations across the data pool generally correspond very well between automatic segmentation data and manual ground truth annotations. The largest deviations are observed for features that rely fundamentally on axial segmentation quality (cell volume, XZ perimeter, DV length, ML-DV anisotropy and DV-AP anisotropy). In these cases, however, manual data annotation itself was frequently problematic (and, in some extreme cases, ambiguous) due to low axial resolution, low image quality as a result of light scattering and aberrations in deeper tissue regions, weak signals from en-face membranes in multilayered tissues and high background levels arising from auto-fluorescence. We note that such limitations in image quality inevitably constrain cell shape quantifications, irrespective of the performance of the image analysis methodology and can only be addressed at the level of the image acquisition process, using advanced microscope designs for improving axial resolution and isotropy (Chhetri et al., 2015).

## Supplemental Tables

**Table S1**, related to Figure 1 | Comparison of processing time for CPU-optimized and GPU-accelerated implementations of RACE

| | **RACE (MS)** | **RACE (NS)** | **RACE (MS, GPU)** | **RACE (NS, GPU)** | **RACE (MS, GPU)*** | **RACE (NS, GPU)*** | **RACE (MS, GPU, Nscale)** | **RACE (NS, GPU, Nscale)** |
|---|---|---|---|---|---|---|---|---|
| **Median filter** | 6.98 | 6.41 | **2.60** | **2.70** | **2.68** | **2.66** | **2.68** | **2.66** |
| **Objectness filter** | 16.45 | 16.41 | **0.92** | **0.86** | **0.80** | **0.79** | **0.80** | **0.79** |
| **Iterative closing** | 11.95 | 12.64 | **1.48** | **1.44** | **1.44** | **1.44** | **1.44** | **1.44** |
| **LoG filter** | - | 3.05 | - | 3.03 | - | 0.77* | - | 3.03 |
| **Distance map** | 1.84 | 1.65 | 1.79 | 1.68 | 0.75* | 0.51* | 1.79 | 1.68 |
| **H-maxima filter** | 12.58 | 11.67 | 12.71 | 11.93 | 3.55* | 2.99* | **2.75** | **2.75** |
| **2D watershed** | 7.74 | 8.16 | 6.88 | 8.48 | 8.85 | 8.24 | **1.38** | **1.38** |
| **Slice-based fusion** | 2.48 | 3.18 | 3.25 | 3.29 | 2.50 | 2.53 | 2.50 | 2.53 |
| **Total** | 66.70 | 70.56 | 36.30 | 40.86 | 27.10 | 27.40 | 19.66 | 23.54 |

Time measurements were performed on the SiMView *Drosophila* data set and are provided in seconds. Bold face entries reflect processing steps that were optimized using CUDA-based computation on a GPU or efficient CPU processing using the C++ library Nscale (Teodoro et al., 2013) to maximize overall pipeline performance. Using GPU acceleration, we achieve a 2.5-fold speedup for the median filter, a 19.5-fold speedup for the Hessian-based objectness filter and an 8.5-fold speedup for the iterative morphological closing. Using CPU-optimized code, we achieve a 4.5-fold speedup for the H-maxima filter and an 8.4-fold speedup for the 2D watershed. A star (*) indicates that the respective processing was performed with the resampled seed detection method using full axial resolution but only half of the lateral resolution (Part 1 of **Supplemental Experimental Procedures**). Total processing time refers to the execution of the entire pipeline, including intensity conversions, image I/O and binary thresholding (which do not reflect time-intensive processing steps but contribute to overall processing time).

**Table S2**, related to Figure 1 | Parameter settings used in the segmentation quality comparison

| **Data set** | *Drosophila* (SiMView) | | Mouse (SiMView) | | *Drosophila* (Confocal) | |
|---|---|---|---|---|---|---|
| **Parameter/Algorithm** | **RACE (MS)** | **RACE (NS)** | **RACE (MS)** | **RACE (NS)** | **RACE (MS)** | **RACE (NS)** |
| **Z vs. XY sampling ratio** | 5 | | | | | |
| **2D median radius** | 2 | | | | | |
| **Binary threshold (MS)** | 0.0008 | – | 0.00004 | – | 0.002 | – |
| **Laplacian-of-Gaussian sigma (NS)** | – | 4 | – | 4 | – | 4 |
| **Binary threshold (NS)** | – | OTSU | – | 0.00002 | – | OTSU |
| **H-maxima level** | 20 | | 40 | 45 | 20 | |
| **H-maxima binarization threshold** | 0.00001 | | | | | |
| **HessianToObjectnessFilter sigma** | 2 | | | | | |
| **HessianToObjectnessFilter beta** | 1 | | | | | |
| **HessianToObjectnessFilter gamma** | 0.1 | | | | | |
| **Morphological closing MinRadius** | 4 | | 1 | | 4 | |
| **Morphological closing MaxRadius** | 4 | | | | | |
| **Morphological closing FilterMask3D** | 1 | | | | | |
| **Slice-by-slice watershed level** | 39 | | 2 | | 50 | |
| **Slice-by-slice RegionProps MinSeedArea** | 3 | | 3 | | 0 | |
| **Slice-by-slice FusionFilter MaxSegmentArea** | 1000 | | 2000 | | 1000 | |
| **Slice-by-slice FusionFilter MinVolume** | 1500 | | 2300 | | 1500 | |
| **Slice-by-slice FusionFilter MaxVolume** | 4000 | | 13000 | | 4000 | |

Parameters shown with dark gray shading are data-dependent and have to be manually adjusted. These parameters were selected for the parameter sensitivity analysis (**Figure S1B-E**). Parameters shown with light gray shading reflect basic properties of the microscope or experiment configuration. Parameters without shading were kept constant across all experiments. However, they can in principle be modified and optimized if required for a particular application. Parameter settings shown here also apply to the (*, SSH), (*, JI) and (*, JI, SSH) benchmarks, which additionally have the respective fusion heuristics enabled. The acronyms used in this table are defined in Tab 7 of **Table S3**.

**Table S3**, related to Figure 1 | Segmentation quality comparison and analysis of cell shape information accuracy

Note: Due to its size, this table is provided as a separate multi-tab Excel file.

## Supplemental Movie Legends

**Movie S1**, related to Figure 2 | Detection of seed points in *Drosophila*

Visualization of seed point detection for a three-dimensional image stack of a membrane- and nuclei-labeled *Drosophila* embryo imaged with SiMView light-sheet microscopy. The movie shows the entire specimen volume slice-by-slice: raw membrane (red) and nuclei (green) image data, binary image of the seeds detected in the nuclear channel, and overlay of the seeds with membrane and nuclear channels, respectively (from left to right).

Note: The movie is provided in QuickTime format. QuickTime and related codecs are freely available at *http://www.apple.com/quicktime/download/*.

**Movie S2**, related to Figure 3 | Cell segmentation in *Drosophila*, zebrafish and mouse embryos

Visualization of segmentation results obtained for three-dimensional image stacks of membrane-labeled *Drosophila* (Part 1), mouse (Part 2) and zebrafish (Part 3) embryos acquired with SiMView light-sheet microscopy as well as a membrane-labeled *Drosophila* embryo acquired with confocal fluorescence microscopy (Part 4). The movie shows the entire specimen volume (Parts 1-3) or the volume of approximately half the embryo (Part 4) slice-by-slice: raw image data (left), a superposition of raw image data and detected segment boundaries (middle) and the cell label image with a randomized color scheme (right).

Note: The movie is provided in QuickTime format. QuickTime and related codecs are freely available at *http://www.apple.com/quicktime/download/*.

**Movie S3**, related to Figure 7 | Tissue anisotropy mapped at the single-cell level in *Drosophila* wild-type and *bnt* mutant embryos

Visualization of tissue anisotropy on the ventral and dorsal sides of *Drosophila* wild-type (Part 1) and *bnt* mutant (Part 2) embryos during early embryonic development. Images were acquired in 15-second (Part 1) and 30-second (Part 2) intervals using SiMView light-sheet microscopy, starting shortly before the onset of gastrulation at 3 hours after egg laying (AEL). The movie shows maximum-intensity projections of background-corrected image data from the ventral and dorsal sides of the embryos (left) as well as the corresponding segmentation results obtained with RACE (right) over 3-hour (Part 1) and 1.5-hour (Part 2) time windows. The segmentation results are visualized using surface rendering. The color code indicates the level of mediolateral-vs.-anteroposterior cell shape anisotropy and was chosen to highlight in particular cell shape changes in response to ventral furrow formation: cells that are elongated along the anteroposterior axis are shown in cyan; uniformly shaped cells are shown in purple; and cells that are elongated along the mediolateral axis are shown in bright magenta. The segmentation data and associated annotation of cell shape anisotropy reveal local changes in cell shape in the course of large-scale tissue reorganization and epithelial folding, including the formation of the ventral furrow, the formation of the cephalic furrow and germ band extension. Calculation and color mapping of anisotropy levels were performed following the procedure described in Part 4 of **Supplemental Experimental Procedures**.

Note: The movie is provided in QuickTime format. QuickTime and related codecs are freely available at *http://www.apple.com/quicktime/download/*.

**Movie S4**, related to Figure 7 | Side-by-side comparison of tissue anisotropy in *Drosophila* wild-type and *bnt* mutant embryos

Side-by-side visualization of tissue anisotropy in wild-type (left) and *bnt* mutant (right) *Drosophila* embryos during early embryonic development. Each embryo is shown from both ventral and dorsal perspectives. Images were acquired in intervals of 20 seconds (wild-type) and 30 seconds (*bnt*) using SiMView light-sheet microscopy, starting shortly before the onset of gastrulation at 3 hours after egg laying (AEL). The movie shows segmentation results that are visualized using surface rendering. The color code indicates the level of mediolateral-vs.-anteroposterior cell shape anisotropy and was chosen to highlight in particular cell shape changes in response to ventral furrow formation: cells that are elongated along the anteroposterior axis are shown in cyan; uniformly shaped cells are shown in purple; and cells that are elongated along the mediolateral axis are shown in bright magenta. This visualization highlights several differences in early cell behavior between wild-type and *bnt* mutant embryos: in particular, shortly after the onset of gastrulation, mediolateral-vs.-anteroposterior cell shape anisotropy near the ventral midline rapidly approaches peak levels in the wild-type embryo, whereas corresponding blastoderm cells in the *bnt* mutant embryo exhibit less pronounced mediolateral cell elongation. Moreover, ML-AP anisotropy relaxes very slowly towards pre-furrow baseline levels after peak anisotropy levels are reached in the *bnt* mutant embryo. Calculation and color mapping of anisotropy levels were performed following the procedure described in Part 4 of **Supplemental Experimental Procedures**.

Note: The movie is provided in QuickTime format. QuickTime and related codecs are freely available at *http://www.apple.com/quicktime/download/*.

**Movie S5**, related to Figure 8 | Joint reconstruction of cell lineages and cell morphology in early *Drosophila* development

Automated reconstruction of cell lineages and cell morphology in an early *Drosophila* embryo using the TGMM algorithm *(Amat et al., 2014)* and RACE, respectively. The imaging experiment underlying this video was performed with a SiMView light-sheet microscope. Imaging started shortly before the onset of gastrulation at 2 hours after egg laying (AEL) and image stacks of both cell nuclei and cell membranes were acquired every 20 seconds. Following image acquisition and multi-view image fusion, cell tracking was performed with TGMM using the fluorescently labeled cell nuclei. Information about segmented nuclei and their temporal associations were then propagated as seed points to the RACE framework. Finally, cell shapes were segmented by RACE using the fluorescently labeled cell membranes. The video shows maximum-intensity projections of background-corrected nuclei and membrane image data for the ventral side of the *Drosophila* embryo (left) as well as the corresponding segmentation results obtained with our segmentation algorithm for both ventral and dorsal sides of the embryo (right). The color code applied to the segmentation results was initialized in the first frame, using a color gradient from anterior to posterior, and was then propagated to subsequent frames using the cell tracking information.

Note: The movie is provided in QuickTime format. QuickTime and related codecs are freely available at *http://www.apple.com/quicktime/download/*.

## Supplemental Software

**Software S1**, related to Figure 1 | RACE cell segmentation framework for Windows, Mac OS X and Ubuntu

This software archive contains our cell segmentation framework RACE compiled for Windows 64-bit (folder “RACE_Windows”), Mac OS X (folder “RACE_MacOSX”) and Ubuntu (folder “RACE_Ubuntu”), including image test data from a membrane- and nuclei-labeled *Drosophila* embryo. The archive also contains the RACE user guide (folder “User_Guide”) and the RACE video tutorial in Flash and QuickTime formats (folder “Video_Tutorial”).

Note: The source code of the RACE framework is hosted as a Git repository and available from *https://bitbucket.org/jstegmaier/race*.

## Supplemental Experimental Procedures

### **Part 1 |** RACE algorithmic design

*Overview and general design principles*

Several algorithms for cell segmentation have been presented in recent years. A common initial step in most methods is the enhancement of locally plane-like structures. Starting with simple approaches such as Gaussian and median filtering for noise removal, gradient-based methods such as edge indicator functions or directional coherence enhancement filters can be used to emphasize cell membrane structures (Adiga et al., 2006; Hodneland et al., 2009; Zanella et al., 2010). More complex approaches are based on the eigensystem of the Hessian matrix of each pixel at different regularization scales, namely Partial Differential Equation (PDE)-based approaches like anisotropic diffusion filtering (Kriva et al., 2010; Perona and Malik, 1990; Pop et al., 2013) or objectness filters that exploit geometrical properties of the eigenvalues to enhance shapes of a specific dimensionality in the images (Frangi et al., 1998; Michelin et al., 2013; Mosaliganti et al., 2012). In the field of *connectomics*, where similar membrane segmentation problems exist, machine learning classifiers, such as random forests or deep neural networks, deliver remarkable membrane reconstruction quality (Andres et al., 2012; Ciresan et al., 2012; Huang and Jain, 2013). However, we did not consider such classification-based approaches as possible solutions to our field of application because of the very high computational cost and the substantial time needed to obtain ground truth for training the classifiers.

In most cases, the edge-enhanced images are still not ideal for automated segmentation because of discontinuities in the cell membrane signal that might be caused by inhomogeneous label expression levels, limited axial resolution or en-face membranes that are oriented parallel to the plane imaged in the microscope (Mosaliganti et al., 2012). To further refine the enhanced cell membrane images and close gaps in the structures, methods such as the viscous watershed transform have been successfully applied (Olivier et al., 2010; Vachier and Meyer, 2005). Another option for closing gaps in the membrane signal is perceptual grouping achieved by tensor voting (Michelin et al., 2013; Mosaliganti et al., 2012) or deformable models adapted to membrane segmentation (Pop et al., 2013). Finally, based on the enhanced membrane image, the actual segments are commonly extracted using topological methods such as the watershed transform (Michelin et al., 2013; Mosaliganti et al., 2012; Olivier et al., 2010), graph-cuts (Funke et al., 2012) or PDE-based methods such as subjective surfaces (Zanella et al., 2010), advective level sets (Mikula et al., 2011) or active meshes (Pop et al., 2013). A common drawback of these methods is the fact that they are computationally expensive because they perform a series of global 3D image processing operations, such as anisotropic diffusion or watershed, in order to reconstruct cell shapes. Many of these methods furthermore require up-scaling of the image data along the z-axis (i.e. the detection axis of the microscope) in order to obtain isotropic resolution for optimal segmentation results. This limits their applicability to the terabytes of anisotropic image data generated per experiment by next-generation fluorescence light microscopy methods.

In this study, we systematically developed a highly efficient image analysis pipeline, termed RACE, which reliably extracts cell shapes directly from anisotropic microscopy images with fluorescently labeled cell membranes. First, we use a combination of median filtering, Hessian-based ridge enhancement and iterative morphological closings to improve imperfect membrane structures in a similar fashion to previous algorithms mentioned above (**Figure S1A**). However, we restrict most of the operations either to 2D slices or to local 3D neighborhoods to avoid costly computational operations. Second, inspired by the automated reconstruction of neural circuits in the field of *connectomics*, we then extract meaningful 2D cell regions from the enhanced membrane image slices and combine these using discrete combinatorial optimization techniques to generate complete 3D cell shape segmentations (Funke et al., 2012; Liu et al., 2014). However, the requirements for segmenting individual cells in an entire embryo are different from those in *connectomics*, where magnification levels are significantly higher and the investigated sub-cellular structures are not compact objects but typically span rather large regions of the acquired volume. We therefore present specialized seeding techniques for the fusion of individual 2D segments using either the membrane images themselves or, optionally, additional cell nuclei image data. Finally, we present two heuristics that further improve segmentation results.
High computational efficiency of all involved processing steps was one of our design priorities, ensuring suitability of the pipeline for large-scale time-lapse data sets of embryonic development. As a result of our strategy to directly process the raw, anisotropic 3D image data produced by the microscope and systematically employ efficient processing operators, we obtain a speed-up of up to two orders of magnitude compared to existing methods, without trading off segmentation quality. The computational framework presented here also consistently achieves higher segmentation accuracy across multiple biological model systems and imaging modalities compared to state-of-the-art methods (**Figures 1B**, **6C**, **S4A-B** and **S7**, Tabs 1-3 of **Table S3**). This combination of high speed and high accuracy enables, for the first time, real-time reconstructions of cell shape dynamics in live imaging experiments at the scale of entire developing embryos comprising up to tens of thousands of cells. In the following, we describe the RACE processing workflow step by step.

*Noise removal and Hessian-based membrane enhancement*

Median filtering represents a simple pre-processing method for eliminating Poisson noise. We apply a 2D median filter to the raw membrane image $I^{mem}$ (**Figure S2A-B**) with a radius that is scaled according to the relative physical spacing of voxels along all image dimensions. This produces the median-filtered image $I^{med}$ (not shown).

We then use a second pre-processing step to enhance membrane structures in the median-filtered image and to further reduce background noise based on specific properties of the eigenvalues of the Hessian matrix at each pixel location. Originally developed to enhance vessel-like structures in 3D images (Frangi et al., 1998), Antiga *et al.* recently presented a generalization of this approach for enhancing M-dimensional shapes in N-dimensional images (Antiga, 2007).

Membrane structures in 3D images correspond to 2D objects, *i.e.* plane-like structures in a local neighborhood, and can therefore ideally be reconstructed using $M = 2$ and $N = 3$ in the generalized objectness function:

$$P(\lambda)_\sigma = \begin{cases} e^{-\frac{R_B^2}{2\beta^2}} \cdot (1 - e^{-\frac{S^2}{2\gamma^2}}) & ,\lambda_3 < 0 \\ 0 & ,\lambda_3 \geq 0 \end{cases} \tag{1}$$

where $|\lambda_1| \leq |\lambda_2| \leq |\lambda_3|$ are the sorted eigenvalues of the Hessian matrix at each pixel location. The eigenvalues are calculated on a Gaussian-smoothed image using a standard deviation $\sigma$ that corresponds to the regularization scale for the Hessian. $S = \sqrt{\lambda_1^2 + \lambda_2^2 + \lambda_3^2}$ is the Frobenius norm of the Hessian matrix, $R_B = |\lambda_2| / |\lambda_3|$ and $\beta, \gamma$ are user-defined weights (Antiga, 2007). The factor containing the $R_A$ expression found in the original formulation is omitted, as it is equal to 1 for the parameterization of $M = 2$ and $N = 3$. In those cases where $|\lambda_3| >> |\lambda_2|$ and $S$ is large, the response of this filter is high, *i.e.* locally plane-like structures such as membranes are enhanced, while noise in background regions is efficiently suppressed if $S$ is close to zero. By evaluating Eq. (1) in a $\sigma$-neighborhood around each voxel, a Hessian-based edge-enhanced image $I^{hee}$ is obtained (**Figure S2C**).

*Iterative morphological closing*

The edge-enhanced image $I^{hee}$ is subsequently processed using an adapted pre-processing step known as the viscous watershed transform (Vachier and Meyer, 2005). This strategy has been successfully applied to membrane segmentation (Olivier et al., 2010); however, previously presented approaches segmented relatively small numbers of cells and required manual interaction to place markers for a seeded watershed-based segmentation. The basic idea of the viscous watershed transform is to iteratively apply morphological operators with increasing radii of the structure element before applying the common watershed transform to the image. This idea can be formalized by the following recursion:

$$I_r^{cl} = \begin{cases} I^{hee} & ,r = 0 \\ I_{r-1}^{cl} \bullet S_r & ,r > 0 \end{cases} \tag{2}$$

where $I_r^{cl}$ is the iteratively closed input image $I^{hee}$ up to radius $r$ and the $\bullet$-operator represents the morphological closing of image $I_{r-1}^{cl}$ with structure element $S_r$.

This pre-processing step simulates the flooding of the topographic landscape with a viscous fluid and helps reducing leakage among disconnected structures, such as locations with poor membrane signal (**Figure S2D**).

In our cell segmentation pipeline, we use an Euclidean sphere as a structure element with radii scaled according to the physical voxel size. Instead of using physically correct modeling of a viscous fluid as proposed by Vachier *et al.* (Vachier and Meyer, 2005), we restrict the range of structure element radii to a predefined range. Throughout the experiments performed in this study, a set of radii $r \in \{1,2,3,4\}$ turned out to be sufficient. The maximum radius can be determined empirically and should be constrained to the size of the smallest compartments that need to be segmented as individual cells. Using larger radii causes over-smoothing of object contours and increases processing time unnecessarily owing to the larger filter kernels. Additionally, we avoid filtering in border regions where the kernel does not fit completely inside the image, in order to improve performance of the used neighborhood iterators. As specimens are typically padded sufficiently well in most imaging experiments, skipping border regions usually does not affect segmentation results.

*Slice-based watershed segmentation*

In our experience, performing 3D watershed segmentation in pre-processed images $I_r^{cl}$ does not produce high-quality results in most cases, as it has a tendency to merge segments and/or lead to "leakage" of cells with membrane gaps into background regions. Even computationally expensive methods such as ACME (Mosaliganti et al., 2012) were not able to segment cells sufficiently well after pre-processing owing to limitations arising from the low axial resolution in some of our tested images. To improve performance of the 3D watershed, existing approaches transform input images to an isotropic representation, *e.g.* using linear interpolation schemes (Michelin et al., 2013; Mosaliganti et al., 2012). However, this up-scaling dramatically increases image size, memory consumption and processing time, and still does not entirely solve the leakage problem. Thus, in order to be able to directly extract segmentation data from anisotropic images, we employ slice-based 2D watershed segmentation with a subsequent fusion step based on discrete combinatorial optimization.

Applying the standard watershed transform in 2D to the pre-processed image $I_r^{cl}$ yields a segmentation image $I^{ws}$ with high 2D segmentation quality in each image slice (**Figure S2E**-**F**), as a direct result of the relatively high lateral resolution in typical microscopy image data sets. In addition, the processing of individual 2D slices can be perfectly parallelized by distributing slices to available CPU cores without a need for communication between concurrent threads.

*Seed detection for initializing the merging of 2D watershed regions across slices*

To initialize the fusion of 2D regions, we employ a seeding approach. Seeds are either single pixels or small 3D segments that should be fully enclosed by single cells, and each seed should have a unique label assigned. Depending on the available image data, RACE provides different methods for seed extraction. In the simplest case, if only the membrane channel is available (**Figure 2A**), the inverted membrane-enhanced image can be used as a pseudo-nuclei channel (**Figure 2B**). Subsequently, the image has to be binarized and connected regions can be separated using a thresholded Euclidean distance map (EDM) on the inverted, binarized pseudo-nuclei image or by extracting regional maxima from the EDM using the H-maxima transform

(Soille, 2003) (**Figure 2C**). We use a fast, linear-time algorithm to calculate the Euclidean distance transform (Maurer Jr et al., 2003). Labeling the connected components of the extracted maxima of the EDM yields the seed image $I^{ms}$ for the seed-based segment fusion (**Figure 2D**).

If nuclear image data is available, RACE can take advantage of this information to improve segmentation quality though a more accurate seed detection. With nuclear image data at hand, seeds can be extracted in a similar way as for membrane image data, but skipping the initial image inversion and directly calculating the EDM and the corresponding maxima from the thresholded, Laplacian-of-Gaussian (LoG) filtered raw image $I^{\log}$ (**Figure 2G-H**). The seed points detected in nuclear image data will be denoted by $I^{ns}$ (**Figure 2I**). We note that in contrast to our previous work on cell nuclei segmentation (Stegmaier et al., 2014), the seeds created by RACE are small 3D segments instead of single pixels, which allows us to directly initialize multiple 2D slices with a single seed label.

For the quality comparisons discussed below we used the extracted H-maxima as seed points. However, if cell sizes are fairly uniform throughout the data set, this step can be substituted by simple binary thresholding of the EDM image to further improve processing speed. If nuclei sizes are heterogeneous but the distance between individual nuclei is sufficiently large, the seed detection step can be performed on a down-sampled version of the respective images to improve speed without trading off accuracy. Even in high-speed light-sheet microscopy imaging experiments, the RACE segmentation framework is capable of real-time performance by either using down-sampled seed detection in combination with our GPU-accelerated code modules (**Figure S4C-D**) or by using our fast CPU-optimized implementation of the H-maxima transform based on the C++ library Nscale (Teodoro et al., 2013).

*Intersection calculation*

Intersections of slice pairs in the 3D label image stack $I^{ws}$ obtained by slice-by-slice application of the watershed transform are identified using a label histogram for each pair of neighboring slices. Considering two successive slices with $N = \max(I_z^{ws})$ labels in slice $z$ and $M = \max(I_{z+1}^{ws})$ labels in slice $z+1$, a label matrix with dimensions $L = (N+1) \times (M+1)$ is created. The size increment in each dimension is due to the background label, which is usually set to zero. By iterating over all pixels in the image, the bins in the 2D histogram are incremented using the labels in $I_z^{ws}$ as a row indicator and the labels $I_{z+1}^{ws}$ as a column indicator. After the histogram has been successfully filled, each non-zero entry of the 2D histogram corresponds to an intersection of two segments. The total number of pixels in the intersection is equal to the bin value. At this stage it is usually helpful to discard very unlikely correspondences based on overlap percentage, minimum size or intensity features. In addition to calculating the intersections of neighboring 2D slices, we calculate the intersections of all 2D segments with the seed image $I^{ms}$ or $I^{ns}$, respectively. Each of the 2D segments that intersects with one or more seeds is labeled with the unique seed label that has maximum overlap with the segment. All 2D segments that are not touching a seed are temporarily labeled with the background label and will be merged with labeled segments in the next step. Moreover, to reduce the amount of false positive detections, small seed segments consisting only of a few pixels can be efficiently suppressed using a minimum size criterion.

*Segment fusion*

Based on the calculated intersections and identified seeds, 2D segments are combined to complete 3D cell shapes (**Figures 1A** and **S3**). For segments that are already intersecting a seed the respective seed label is assigned. In the next step, all intersections that contain a labeled segment are added to a queue, which is sorted by intersection similarity in descending order. The intersection similarity for two sets of pixels $A, B$ can be based on, for instance, the Jaccard index ($A \cap B / A \cup B$), the minimum relative overlap ($\min(A \cap B / A, A \cap B / B)$) or on a classification score of a trained intersection classifier. Here we used the Jaccard index, since it (1) performs well in practice, (2) can be directly obtained from the intersection histogram and (3) does not require any training for parameter tuning. Starting with the highest-scoring intersection, the respective seed label of the labeled segment is propagated to the unlabeled segment. For each newly labeled segment, its intersections with unlabeled segments are inserted into the sorted queue using a binary search. Processed intersections are removed from the queue and the fusion process continues until the queue is empty. This fusion procedure is very efficient since it operates in a sparse graph connecting 2D watershed regions across slices.

*Fusion heuristics*

In the case of perfect seeding, the number of detected cells should match the ground truth. However, in a more realistic scenario, such as when dealing with elongated cell or nuclei shapes, the seeding approach likely produces multiple seeds for a single cell, yielding split cells in the final segmentation (**Figure S3C**). We thus use two post-processing heuristics to further improve cell segmentation results.

The first correction heuristic uses a minimum-spanning tree (MST) obtained by a clustering of 2D segments similar to Kruskal's algorithm, based on an intersection similarity measure such as the Jaccard index or the minimum relative overlap. In contrast to the seeded fusion described in the previous section, new nodes are introduced to the segmentation tree by using the most similar intersections (**Figure S3D**). Furthermore, we prohibit merges that would introduce a local extremum in the mean intensity profile along the fused segments. This rule is based on our prior knowledge that no true cell should contain an intermediate local maximum in the membrane mean intensity feature or a local minimum for the nucleus mean intensity feature. In cases where both parts of an intersection are already labeled, *i.e.* where the segments are part of two separate nodes in the segmentation tree, we allow further merging only if no local extremum in the mean intensity profile along the merged segment would be introduced. The actual correction of the seed-based segmentation approach is performed by comparing the segments of the similarity-based fusion to the segments of the seed-based fusion. The similarity-based fusion has a strong tendency to over-segmentation. However, it minimizes under-segmentation due to the local extremum rule. If a single segment in the similarity-based fusion matches two or more segments in the seed-based fusion, this is a strong indicator that the segments of the seed-based fusion should be combined into a single segment.

Similar to existing methods (Fernandez et al., 2010; Khan et al., 2014), we employ a second heuristic based on a minimum volume criterion, *i.e.* we take into consideration that it is unlikely that correctly detected cells comprise only a few voxels. To fix such over-segmentation errors, we further merge cell fragments with the most likely match in their local neighborhood using the available intersection information. Depending on the expected range of cell volumes in the investigated biological specimen, a minimum expected volume can easily be determined. To prevent erroneous merging by this heuristic, two segments are only fused if the new segment is still smaller than a predefined maximum size. Unlike existing methods, our heuristics are not operating on the image level but exclusively on already extracted segmentation data. Thus, the time required for this refinement step is almost negligible compared to the other modules of the RACE algorithm.

### *Implementation details*

The membrane segmentation pipeline was implemented in C++ using the Insight Toolkit SDK (*http://itk.org/*) and the Qt SDK (*http://qt-project.org/*). We successfully compiled and tested the software on multiple operating systems, including Windows 7, Ubuntu 12.04 LTS, Scientific Linux 5.8 and Mac OS X 10.9.3. We also provide high-performance GPU-accelerated implementations of RACE based on CUDA (*https://developer.nvidia.com/cuda-zone*) to further optimize processing speed and eliminate memory bottlenecks, specifically for median filter, Hessian-based ridge enhancement and iterative morphological closing. The GPU implementations can be enabled optionally if hardware requirements are met. Furthermore, we make use of the C++ library Nscale (Teodoro et al., 2013), which depends on the OpenCV SDK (*http://opencv.org*), to speed up the CPU-based watershed transform and the extraction of H-maxima. In order to make the RACE computational framework easily accessible to the community, we provide precompiled executables for several major operating systems as well as a graphical user interface for easy parameter adjustment and data processing. Our open-source implementation is licensed under the GNU-GPL and is freely available for download at *https://bitbucket.org/jstegmaier/race.*

## **Part 2 |** Performance evaluation

### *Ground truth generation and validation procedure*

We manually annotated representative image regions of SiMView light-sheet microscopy recordings of *Drosophila* and mouse embryos and confocal fluorescence microscopy image data of a *Drosophila* embryo. Dense ground truth segmentations were created using the Fiji plugin TrackEM2 to label 2D segments (Cardona et al., 2012) and we developed a custom-made MATLAB user interface for the manual fusion of the resulting 2D segments. Owing to the substantial efforts involved in generating manual annotations as well as limitations with respect to the image size that other algorithms can handle, images were cropped to smaller regions containing ~50-100 cells for annotation. We then processed several of these smaller image regions from various locations within each specimen. The values listed in the result tables represent average performance over all image regions in each image stack. In order to avoid errors resulting from edge effects in border regions, we first segmented larger image regions and then cropped the result to the respective regions selected for quality assessment, under consideration of the maximum filter kernel diameter of all involved processing operators. Finally, since ACME produced many tiny segments at the membrane junctions, we also added a size constraint, *i.e.* only segments larger than 300 voxels were considered in the quality comparison.

### *Validation measures*

For a quantitative comparison of segmentation quality of our algorithm and other existing methods, we used the measures proposed by Coelho *et al.* (Coelho et al., 2009): Rand Index (RI), Jaccard Index (JI), Normalized Sum of Distances (NSD) and Hausdorff Metric (HM). The Rand Index measures the fraction of pixel pairs where labels produced by automatic segmentation and manual annotation agree (higher values are better). Similar to the RI, the Jaccard Index measures the ratio of matching pixel pairs to non-matching pixel pairs (higher values are better). The Hausdorff Metric is defined as the maximum of the set of minimal distances of two compared shapes (lower values are better) (Bamford, 2003). Finally, the Normalized Sum of Distances reflects the average distance of labeled pixels that do not agree between automated segmentation and manually annotated reference image (lower values are better) (Coelho et al., 2009). Moreover, we distinguished between added, missing, split and merged segments among the topological errors in the automatic segmentation. We used this information to calculate precision and recall by considering added and split cells as false positives and merged and missing cells as false negatives.

### *Selection of comparative methods*

We compared our algorithm to three different state-of-the-art cell segmentation methods. First, we compared it to the ACME method proposed by Mosaliganti *et al.* This method uses Hessian-based ridge enhancement with tensor voting for closing remaining gaps in the membrane signal, followed by a watershed transform on the distance map of the inverted, thresholded tensor voting image (Mosaliganti et al., 2012). The second approach (MARS) proposed by Fernandez *et al.* uses a framework based on alternate sequential filtering for pre-processing and a size-dependent iterative 3D watershed transform for actual segmentation (Fernandez et al., 2010). The third method, EDGE4D by Khan *et al.*, uses a combination of histogram-based contrast adjustment, Difference-of-Gaussian filtering, 3D rank filters for edge enhancement and a watershed segmentation that is guided by morphological criteria for cells (Khan et al., 2014). This method optionally makes use of an additional nuclear channel to optimize the segmentation results. In contrast to our algorithm, these other methods provide high-quality segmentation results only if image data are first up-scaled to isotropic 3D spatial sampling, which for typical light microscopy data sets artificially increases image size by a factor of 5 to 10. Therefore, in order to ensure a fair comparison of the different methods with respect to the actual image size used in each scenario, we determined the processing speed in voxels per second, in addition to reporting the required total processing time in seconds. An overview of the acronyms used to refer to the various methods included in this comparison as well as a detailed list of the parameter settings employed for each data set are provided in Tab 7 of **Table S3** and in **Table S2**, respectively.

For all methods, we manually fine-tuned parameters in order to produce and report optimal results. In most cases, extensive parameter search methods were not applicable owing to restrictions imposed by graphical user interfaces or the substantial processing time required for algorithm execution.

*Assessment of cell segmentation quality*

The segmentation quality measures for the SiMView *Drosophila* dataset are listed in Tab 1 of **Table S3**. Our implementation produced the best results with respect to RI, NSD and HM measures. Regarding recall values, RACE (NS, *) achieved the best results, closely followed by the ACME method. However, ACME failed to provide similarly high precision values and produced multiple false positive detections in background regions. The highest precision values were obtained by EDGE4D, RACE (MS, *), RACE (NS, SSH) and RACE (NS, JI, SSH). Although MARS produced only a few splits and missing cells, many cells were merged and added to the segmentation results, resulting in low recall and precision values for this data set. Overall, using an additional nuclear channel clearly helped with achieving high recall and precision values, as indicated by the results obtained with RACE (NS, *) and EDGE4D.

The mouse data set (Tab 2 of **Table S3**) has higher diversity in cell shapes and represents a more challenging scenario for all algorithms. The values obtained for RI, JI and HM measures were comparable for all investigated methods. Interestingly, the additional nuclear channel did not help improve results for this data set. This is due to the fact that nuclei are more densely packed in the mouse embryo data set and that it is hard to perform optimal cell nuclei detection, which affects the seeding for membrane segmentation. Even during visual inspection of the images by a human, there was uncertainty in deciding on the exact nucleus location in some cases.

Presumably, this effect led to the slightly increased NSD values for RACE (NS, *). The best recall value was achieved by ACME, closely followed by RACE (MS) and RACE (MS, JI). The best precision values for this data set were also achieved using membrane-based seeding techniques, namely RACE (MS, *), MARS and ACME. The small segment heuristic efficiently increased the precision of RACE (MS, SSH) and RACE (MS, JI, SSH) by 0.06 points.

The confocal data set of an early *Drosophila* embryo (Tab 3 of **Table S3**) contained a higher noise level than its SiMView counterpart. This caused the seed detection method used by MARS to miss many cell centroids, which resulted either in many merged regions or in many missed cells depending on parameter settings. Despite the exclusion of small segments caused by the junction issue of the ACME method, the precision of 0.77 was not satisfactory for this data set. Furthermore, it only reached a recall value of 0.70, which was mainly caused by background signal leakage and several merged cells. As nuclei were again nicely separated in this data set, nuclei-based seed detection methods delivered the best results. EDGE4D as well as each of our nuclei-based versions of RACE provided the highest scoring results with respect to both precision and recall. As the initial seeding of RACE (NS) was already almost perfect, the fusion heuristics did not further improve results in this case and even slightly increased the number of merged nuclei (RACE (NS, SSH), RACE (NS, JI, SSH)). In contrast, the precision of RACE (MS) greatly benefited from the small segment fusion heuristic (RACE (MS, SSH), RACE (MS, JI, SSH)), which increased precision by almost 20 percent. Although RACE (MS, JI) did not show a noticeable improvement, the combination of both heuristics RACE (MS, JI, SSH) delivered the best results among the membrane-based seed detection methods.

In all scenarios, the highest F-score (geometric mean of precision and recall) was always achieved by RACE. In addition to the quantitative comparison, an exemplary qualitative comparison of segmentation accuracy in images of a *Drosophila* embryo is provided in **Figure 4**. Furthermore, the quantitative segmentation quality of all algorithms across all investigated scenarios is graphically summarized in **Figure S4A-B**.

*Performance comparison of the investigated approaches*

For all data sets presented here, RACE was the fastest by a large margin. Processing time increased by 6 % on average when using an additional nuclear channel for seed detection owing to the additional Laplacian-of-Gaussian filtering and read operations (**Figure S4C-D**, Tabs 1-3 of **Table S3**). Although MARS provided high processing speeds when applied directly to the anisotropic images, it produced poor results without image up-scaling (data not shown). Using differently sized regions of an early *Drosophila* embryo, we showed that our method scales linearly with the number of voxels in the image data, *i.e.* it is well suited even for larger specimens such as entire mouse or zebrafish embryos (**Figure S4C-D**). Most other investigated methods also showed a linear relationship between processing time and voxel count in the image data. However, their use of global 3D image processing algorithms, such as watershed and tensor voting, and their dependency on isotropic image data to obtain accurate results increased processing time by up to two orders of magnitude compared to our algorithm. ACME also required a large amount of memory, which precluded processing even of relatively small image regions on a computer workstation with 128 GB of RAM. The results shown in **Figure S4C-D** and **Table S1** indicate that our GPU-accelerated version of RACE efficiently eliminates some of the main bottlenecks of the pipeline, such as median filtering, Hessian-based objectness filtering and iterative morphological closing. Moreover, our CPU-optimized code for H-maxima filtering and 2D watershed further accelerates image processing and eliminates the two remaining bottlenecks. The combined performance improvements achieved with our GPU and Nscale modules allows us to segment a *Drosophila* embryo at the full spatial resolution provided by the SiMView light-sheet microscope (1316 x 628 x 111 voxels, 16-bit depth, 175 MB) in less than 20 seconds. Our method therefore allows real-time analyses of developing model organisms, even in the context of high-speed imaging with state-of-the-art light-sheet microscopy. All performance measurements, except for the EDGE4D benchmarks, were performed on a computer workstation with two Intel Xeon E5 CPUs at 3.1 GHz (16 cores in total), 196 GB RAM, NVidia Tesla K20 GPU and Windows 7 Professional 64-bit. EDGE4D was tested on an Apple MacBook Pro with Intel Core i7 CPU at 2.3 GHz (4 cores in total), 16 GB RAM and Mac OS X 10.9.5. We also tested EDGE4D on a high-end Mac Pro with two Intel Xeon E5 CPUs at 2.7 GHz but found that processing time did not change significantly, since very few operations are multi-threaded and time spent on I/O is minimal. Thus, processing time essentially depends only on the performance of a single CPU core in this case.

**Part 3 |** Instructions for using the RACE segmentation framework

*Step-by-step protocol, troubleshooting guide and RACE video tutorial*

We provide text and video materials that explain and demonstrate how the RACE cell segmentation framework can be applied to new image data, including a detailed step-by-step protocol (Box 1 in the RACE user guide provided in **Software S1**), a troubleshooting guide (Box 2 in the RACE user guide provided in **Software S1**) and a RACE video tutorial (folder "Video_Tutorial" in **Software S1**).

*Building the framework from the sources*

The C++ implementation of the RACE segmentation algorithm is available from *http://www.bitbucket.org/jstegmaier/race/.* We take advantage of the CMake build tool to generate project files for various operating systems that make it easy to work with the compiler of your choice (*http://www.cmake.org/*). This tool is used to configure the Insight Toolkit (ITK) libraries and to generate the related project files for both ITK and the segmentation executable itself. ITK is freely available for download from *http://www.itk.org/*. Moreover, the Qt libraries are required and can be obtained from *http://qt-project.org/* (specifically, the QtCore and QtWidgets modules are required). Detailed installation instructions for ITK and Qt are provided on the respective webpages. For development and software testing, we used CMake v3.0.0, ITK v4.3 and Qt v5.2 under Windows 7 Professional 64-bit using Microsoft Visual Studio 2012 and its associated C++ compiler. The software has also been successfully compiled and tested under Windows 8.1, Ubuntu 12.04 LTS, 14.04 LTS, Scientific Linux 5, 6 and Mac OS X 10.9.3.

Notes:

- For processing TIFF files larger than 4GB it is necessary to have a BigTIFF-compatible `libtiff` version installed and to enable the ITK CMake flag `ITK_USE_64BIT_IDS` during ITK makefile generation (see *http://bigtiff.org/*). Furthermore, the flag `ITK_USE_REVIEW` needs to be enabled.
- For processing large images, ITK and Qt libraries and executables need to be compiled as 64-bit versions.
- When observing errors related to missing Qt headers or libraries, check if the header and library paths have been properly set by CMake. Otherwise, add "`QTDIR/include/`", "`QTDIR/include/QtCore/`" to the header search path and "`QTDIR/lib/`", "`QTDIR/bin/`" to the library search path. Furthermore, confirm that "`qtmain.lib`" and "`Qt5Core.lib`" are listed as additional dependencies in the Visual Studio project linker settings.
- On Windows systems, we recommend installing ITK at most two sub-folders away from the system root. Otherwise path name length limits of the file system may be exceeded.

- For faster compilation of ITK the following options can be disabled: `BUILD_EXAMPLES`, `BUILD_DOCUMENTATION`, `BUILD_SHARED_LIBS` and `BUILD_TESTING`.
- If you would like to use the GPU-accelerated version of the pipeline, the CUDA Toolkit is needed as well (*https://developer.nvidia.com/cuda-toolkit*).

*Compiling the sources*

After installation and compilation of all prerequisites, it should be possible to compile the application. This can be done with the CMake build tool and a compiler of your choice using the `CMakeLists.txt` located in the folder `PROJECTROOT/Project/CMakeQt5`. Therefore, the source path of CMake has to be set to `PROJECTROOT/Project/CMakeQt5/` and the build path has to be set *e.g.* to `PROJECTROOT/Project/Buildx64/` (folder names are denoted relative to the installation directory). If ITK, Qt or CUDA are not found automatically, make sure to redirect CMake to the paths the respective libraries are located at. After successful Makefile generation using CMake it should be possible to compile the segmentation algorithm, *e.g.* using the `make` command within a Unix terminal or building the generated Visual Studio project files.

*Application example*

Once code generation is complete and the executable has been successfully built, the program can be started via the Unix terminal application with the call `./XPIWIT < input.txt` or in the Windows command prompt using `XPIWIT.exe < input.txt`, where `input.txt` is a text file that determines (1) the input and output parameters for the executable, (2) an XML pipeline to process and (3) additional parameters. Note that the executable itself is called without parameters but all inputs are piped either directly or using a file. The application expects information about the following input parameters in the `input.txt` text file:

```
--output PROJECTROOT/Example/Results/
--input 0, PROJECTROOT/Example/Data/Membrane/Drosophila_c=00_t=0010.tif, 3, float
--input 1, PROJECTROOT/Example/Data/Nuclei/Drosophila_c=02_t=0010.tif, 3, float
--xml PROJECTROOT/Example/membraneSegmentationDrosophilaMS.xml
--seed 0
--lockfile off
--subfolder filterid, filtername
--outputformat imagename, filtername
--end
```

The most important parameters are the output path (line 1), the input paths (lines 2 and 3) and the path of the XML file (line 4). Always use “/” as a folder separator instead of “\”, even on Windows systems. The remaining parameters can be left unchanged. The `Example` folder contains three small 3D data sets showing labeled nuclei and membranes in a *Drosophila* embryo. Input files for the compiled executable as well as XML pipeline files for segmentation and for GPU version are also provided. Make sure to adjust the absolute paths within the input files according to the specific location on your disk. If program execution was successful, the specified output folder should contain a log file including processing time, parameters and pipeline components as well as the resulting image data. All parameters can be adjusted in the file `PROJECTROOT/Example/membraneSegmentationDrosophila*.xml` using a simple text editor. A comprehensive list with available filter options as well as the description of parameters can be requested with the call `XPIWIT.exe --filterlist myfilters.xml` on Windows and `./XPIWIT --filterlist myfilters.xml` on Unix-based systems, respectively.

**Part 4 |** Visualization and analysis of tissue anisotropy in *Drosophila* gastrulation

In order to analyze and visualize cell shape changes during ventral furrow formation in *Drosophila* gastrulation, a directed shape anisotropy measure was computed. After image segmentation with RACE using nuclei-based seeding and data correction with fusion heuristics, we fitted a 3D ellipsoid to the outer shell of the segmented embryo. We calculated the width $w$, height $h$ and depth $d$ of each cell by individually projecting each segment onto the mediolateral tangent, the anteroposterior tangent and the local normal vector of the ellipsoid (corresponding to the dorsoventral axis at locations close to the ventral furrow), respectively. Tangents were calculated at the location of each cell centroid projected onto the ellipsoid surface. Using the resulting cell dimensions, we calculated the directed shape anisotropy measure as follows:

$$Anisotropy = \begin{cases} (w/h)-1, & w/h>1 \\ 1-(h/w), & otherwise \end{cases} \tag{3}$$

Following this definition, an anisotropy value of 0 corresponds to identical cell width and cell height, whereas negative values indicate an elongation along the anteroposterior axis and positive values indicate an elongation along the mediolateral axis. This measure is thus particularly well suited to follow cell shape changes related to ventral furrow formation, since sensitivity is optimized for shape deformations parallel or perpendicular to the ventral furrow. In order to optimize contrast in the visualization of anisotropy values across the embryo shown in **Figures 6A**, **7B**, **S6** and **Movies S3-S4**, we restricted color-coding of anisotropy values to the range [−1; 1.5]. A quantitative analysis of changes in cell shape anisotropy, cell dimensions and cell volume as a function of time was carried out by analyzing segmentation data in an 80-µm-wide corridor along the entire length of the ventral furrow ("ROI 1" in **Figure 6B**) and a $50 \times 50 \times 60$ µm$^3$ sub-volume in wild-type and *bnt* mutant *Drosophila* embryos (green rectangle in **Figure 7B**).

Moreover, complementing the general segmentation quality metrics defined in the previous section, we also quantitatively assessed the accuracy of RACE segmentation data in the specific context of tissue invagination and related changes in cell shape anisotropy patterns at the whole-embryo level. To this end, we manually segmented a representative section of the ventral furrow of a *Drosophila* embryo ("ROI 2" in **Figure 6A**, green rectangle in **Figure S6**). All cells in this region were binned according to the centroid location along the mediolateral axis, using a bin size of 50 pixels (20 µm). To prevent false positive detections in interior regions of the embryo from introducing a possible bias in our measurements and side-by-side performance comparison shown in **Figure 6C**, we only included automatically detected segments overlapping with ground truth segments. Owing to the substantial computational requirements of some of the methods included in the comparison (ACME, EDGE4D, MARS), the large number of adjustable parameters (EDGE4D) and the limited automation capabilities inherent to the use of a graphical user interface (EDGE4D), framework parameters were optimized manually using small image regions of the selected test image and subsequently used for whole-embryo segmentation. Manual parameter optimization of ACME, MARS and EDGE4D was initialized using the default parameters recommended in the respective descriptions of the algorithms. Parameters were then optimized for the test image data by systematically inspecting intermediate results produced in each processing step.

To assess segmentation quality over time in both wild-type and *bnt* embryos, we manually labeled 11 representative regions in both data sets (containing at least 20 cells each), extracted parameter distributions for the cell shape features outlined above and compared median values and standard deviations of these distributions for automatic reconstructions performed with RACE and ground truth annotations (**Figures 7**, **S7**, **Movies S3-S4**).

## Supplemental References

Adiga, U., Malladi, R., Fernandez-Gonzalez, R., and de Solorzano, C.O. (2006). High-throughput analysis of multispectral images of breast cancer tissue. IEEE Trans Image Process *15*, 2259-2268.

Amat, F., Lemon, W., Mossing, D.P., McDole, K., Wan, Y., Branson, K., Myers, E.W., and Keller, P.J. (2014). Fast, accurate reconstruction of cell lineages from large-scale fluorescence microscopy data. Nat Methods *11*, 951-958.

Andres, B., Koethe, U., Kroeger, T., Helmstaedter, M., Briggman, K.L., Denk, W., and Hamprecht, F.A. (2012). 3D segmentation of SBFSEM images of neuropil by a graphical model over supervoxel boundaries. Med Image Anal *16*, 796-805.

Antiga, L. (2007). Generalizing vesselness with respect to dimensionality and shape. Paper presented at: MICCAI.

Bamford, P. (2003). Empirical comparison of cell segmentation algorithms using an annotated dataset. Paper presented at: ICIP.

Cardona, A., Saalfeld, S., Schindelin, J., Arganda-Carreras, I., Preibisch, S., Longair, M., Tomancak, P., Hartenstein, V., and Douglas, R.J. (2012). TrakEM2 software for neural circuit reconstruction. PLoS ONE *7*, e38011.

Chhetri, R.K., Amat, F., Wan, Y., Hockendorf, B., Lemon, W.C., and Keller, P.J. (2015). Whole-animal functional and developmental imaging with isotropic spatial resolution. Nat Methods.

Ciresan, D.C., Giusti, A., Gambardella, L.M., and Schmidhuber, J. (2012). Deep neural networks segment neuronal membranes in electron microscopy images. Paper presented at: NIPS.

Coelho, L.P., Shariff, A., and Murphy, R.F. (2009). Nuclear segmentation in microscope cell images: a hand-segmented dataset and comparison of algorithms. Paper presented at: ISBI.

Fernandez, R., Das, P., Mirabet, V., Moscardi, E., Traas, J., Verdeil, J.L., Malandain, G., and Godin, C. (2010). Imaging plant growth in 4D: robust tissue reconstruction and lineaging at cell resolution. Nat Methods *7*, 547-553.

Frangi, A.F., Niessen, W.J., Vincken, K.L., and Viergever, M.A. (1998). Multiscale vessel enhancement filtering. Paper presented at: MICCAI.

Funke, J., Andres, B., Hamprecht, F.A., Cardona, A., and Cook, M. (2012). Efficient automatic 3D-reconstruction of branching neurons from EM data. Paper presented at: CVPR.

Hodneland, E., Bukoreshtliev, N.V., Eichler, T.W., Tai, X.-C., Gurke, S., Lundervold, A., and Gerdes, H.-H. (2009). A unified framework for automated 3D segmentation of surface-stained living cells and a comprehensive segmentation evaluation. IEEE Trans Med Imag *28*, 720-738.

Huang, G.B., and Jain, V. (2013). Deep and wide multiscale recursive networks for robust image labeling. arXiv, 1310.0354.

Khan, Z., Wang, Y.C., Wieschaus, E.F., and Kaschube, M. (2014). Quantitative 4D analyses of epithelial folding during Drosophila gastrulation. Development *141*, 2895-2900.

Kriva, Z., Mikula, K., Peyrieras, N., Rizzi, B., Sarti, A., and Stasova, O. (2010). 3D early embryogenesis image filtering by nonlinear partial differential equations. Med Image Anal *14*, 510-526.

Liu, T., Jones, C., Seyedhosseini, M., and Tasdizen, T. (2014). A modular hierarchical approach to 3D electron microscopy image segmentation. J Neurosci Meth *226*, 88-102.

Maurer Jr, C.R., Qi, R., and Raghavan, V. (2003). A linear time algorithm for computing exact Euclidean distance transforms of binary images in arbitrary dimensions. PAMI *25*, 265-270.

Michelin, G., Guignard, L., Fiuza, U.-M., and Malandain, G. (2013). Embryo cell membranes reconstruction by tensor voting. Paper presented at: ISBI.

Mikula, K., Peyriéras, N., Remešíková, M., and Stašová, O. (2011). Segmentation of 3D cell membrane images by PDE methods and its applications. Comput Biol Med *41*, 326-339.

Mosaliganti, K.R., Noche, R.R., Xiong, F., Swinburne, I.A., and Megason, S.G. (2012). ACME: automated cell morphology extractor for comprehensive reconstruction of cell membranes. PLoS Comp Biol *8*, e1002780.

Olivier, N., Luengo-Oroz, M.A., Duloquin, L., Faure, E., Savy, T., Veilleux, I., Solinas, X., Debarre, D., Bourgine, P., Santos, A.*, et al.* (2010). Cell lineage reconstruction of early zebrafish embryos using label-free nonlinear microscopy. Science *329*, 967-971.

Perona, P., and Malik, J. (1990). Scale-space and edge detection using anisotropic diffusion. PAMI *12*, 629-639.

Pop, S., Dufour, A.C., Le Garrec, J.-F., Ragni, C.V., Cimper, C., Meilhac, S.M., and Olivo-Marin, J.-C. (2013). Extracting 3D cell parameters from dense tissue environments: application to the development of the mouse heart. Bioinformatics *29*, 772-779.

Soille, P. (2003). Morphological image analysis: principles and applications (Springer).

Stegmaier, J., Otte, J.C., Kobitski, A., Bartschat, A., Garcia, A., Nienhaus, G.U., Strähle, U., and Mikut, R. (2014). Fast segmentation of stained nuclei in terabyte-scale, time resolved 3D microscopy image stacks. PLoS ONE *9*, e90036.

Teodoro, G., Pan, T., Kurc, T.M., Kong, J., Cooper, L.A., Podhorszki, N., Klasky, S., and Saltz, J.H. (2013). High-throughput Analysis of Large Microscopy Image Datasets on CPU-GPU Cluster Platforms. IPDPS / International Parallel and Distributed Processing Symposium IPDPS *2013*, 103-114.

Vachier, C., and Meyer, F. (2005). The viscous watershed transform. J Math Imaging Vis *22*, 251-267.

Zanella, C., Campana, M., Rizzi, B., Melani, C., Sanguinetti, G., Bourgine, P., Mikula, K., Peyrieras, N., and Sarti, A. (2010). Cells segmentation from 3D confocal images of early zebrafish embryogenesis. IEEE Trans Image Process *19*, 770-781.