

# UC San Diego

## UC San Diego Electronic Theses and Dissertations

### Title

The Automated Reconstruction and Analysis of High Resolution Spatial Models of Neuronal Microanatomy

### Permalink

<https://escholarship.org/uc/item/5c04v3hw>

### Author

Perez, Alexander Joseph

### Publication Date

2014

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA, SAN DIEGO

The Automated Reconstruction and Analysis of High Resolution Spatial Models of  
Neuronal Microanatomy

A Dissertation submitted in partial satisfaction of the requirements  
for the degree Doctor of Philosophy

in

Bioengineering

by

Alexander Joseph Perez

Committee in charge:

Professor Mark H. Ellisman, Chair  
Professor Gabriel A. Silva, Co-Chair  
Professor Katerina Akassoglou  
Professor Andrew D. McCulloch  
Professor Gina E. Sosinsky

2014



Copyright

Alexander Joseph Perez, 2014

All rights reserved

The Dissertation of Alexander Joseph Perez is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

---

---

---

---

Co-Chair

---

Chair

University of California, San Diego

2014

## DEDICATION

*To my family,*

*Thank you for the support and guidance.*

*To the memory of my cousins, Andrew and Brian,*

*Thank you for the inspiration.*

## EPIGRAPH

The voyage of the best ship is a zigzag line of a hundred tacks.

– Ralph Waldo Emerson

## TABLE OF CONTENTS

Signature Page .....	iii
Dedication .....	iv
Epigraph .....	v
Table of Contents .....	vi
List of Abbreviations .....	ix
List of Figures .....	xi
List of Tables .....	xiv
Acknowledgements .....	xi
Vita .....	xviii
Abstract of the Dissertation .....	xxi
Chapter 1. Methods for Whole Cell Imaging at High Resolution and their Applications in the Neurosciences .....	1
1.1. Introduction.....	2
1.2. Light microscopy.....	2
1.3. Serial section transmission electron microscopy.....	5
1.4. Serial block-face scanning electron microscopy .....	8
1.5. Focused ion beam scanning electron microscopy .....	14
1.6. Electron tomography .....	17
1.7. Array tomography.....	21
1.8. Discussion and future perspectives .....	23
Chapter 2. The Automatic Segmentation of Multi-scale Neuroanatomical Features in 3D EM Image Stacks .....	28
2.1. Introduction .....	29
2.1.1. The manual segmentation bottleneck .....	29
2.1.2. Automatic segmentation algorithms and their applications in the neurosciences .....	41
2.2. Methods Development and Results .....	48
2.2.1. Data collection and pre-processing .....	48
2.2.1.1. Tissue processing and SBEM image stack acquisition .....	48
2.2.1.2. SBEM stack alignment .....	49
2.2.1.3. Image downsampling and conversion .....	50
2.2.1.4. Histogram equalization .....	54
2.2.2. Pixel classification .....	59
2.2.2.1. Generation of training labels .....	59
2.2.2.2. Training organelle-specific classifiers .....	60
2.2.2.3. Computation of probability maps .....	62
2.2.2.4. Assessment of classifier performance .....	64

	2.2.2.5. Binarization of probability maps .....	66
	2.2.2.6. Comparison to a previously published algorithm .....	70
	2.2.2.7. The impact of image downsampling on automatic segmentation performance .....	79
	2.3. Discussion .....	79
Chapter 3.	From Pixels to Structures: Constructing Models of Neuronal Microanatomy .....	85
	3.1. Introduction .....	86
	3.1.1. Nuclear structure and function .....	88
	3.2. Methods Development and Results .....	94
	3.2.1. The multiplane automatic segmentation algorithm ..	94
	3.2.1.1. Description of the algorithm .....	95
	3.2.1.2. Implementation and Results .....	96
	3.2.2. Interslice interpolation of 3D objects .....	98
	3.2.3. Contour and mesh generation .....	106
	3.2.4. Tools for the automatic analysis of nuclear morphology .....	123
	3.2.4.1. Nuclear and nucleolar morphology and positioning .....	123
	3.2.4.2. Advanced metrics for characterizing nuclear topology .....	131
	3.2.5. Delineation of individual neuronal compartments ....	141
	3.3. Discussion .....	161
Chapter 4.	Chronomorphological Studies of the Mammalian Suprachiasmatic Nucleus .....	164
	4.1. Introduction .....	165
	4.1.1. Neuroanatomy of the suprachiasmatic nucleus .....	172
	4.1.2. A history of biological chronomorphology .....	175
	4.2. Methods .....	185
	4.2.1. Tissue processing and SBEM imaging .....	185
	4.2.2. The subcellular chronomorphology of SCN neurons studied by SBEM .....	186
	4.2.2.1. Nuclei and nucleoli.....	186
	4.2.2.2. Mitochondria .....	187
	4.2.2.3. Stigmoid bodies .....	188
	4.2.3. Electron tomography of SCN organelles .....	192
	4.2.3.1. Serial section electron tomography of the stigmoid body .....	192
	4.2.3.2. Electron tomography of neuronal nuclei...	193

4.3.	Results .....	194
4.3.1.	Nuclear volume, surface area, and topology .....	194
4.3.2.	Nucleolar number and volume .....	195
4.3.3.	Stigmoid body number and morphology .....	195
4.4.	Discussion .....	196
Chapter 5.	Conclusions and Future Perspectives .....	212
5.1.	Contributions, significance, and limitations .....	213
5.2.	Future perspectives .....	215
Appendix A.	A Survey of the Quantitative Methods Used in Published SBEM Studies .....	227
Appendix B.	Training Images and Labels .....	233
Appendix C.	Source Code .....	254
Appendix D.	Example Output from Automated Nuclear Analysis .....	326
References	.....	336

## LIST OF ABBREVIATIONS

3V	Third ventricle
ANOVA	Analysis of variance
AT	Array tomography
ATUM	Automatic tape-collecting ultramicrotome
AuNP	Gold nanoparticle
AVP	Arginine vasopressin
BSA	Bovine serum albumin
BSE	Backscattered electron
CH	Convex hull
CHD	Convex hull difference
CHM	Cascaded hierarchical model
CLAHE	Contrast limited adaptive histogram equalization
CLEM	Correlated light and electron microscopy
CPD	Coherent point drift
CSV	Comma-separated values
CT	Computed tomography
cTEM	Conventional transmission electron microscopy
DAB	Diaminobenzadine
DCE	Discrete curve evolution
DM	Dorsomedial
dSTORM	Direct stochastic optical reconstruction microscopy
ESEM	Environmental scanning electron microscope
EHS	Exact histogram specification
ER	Endoplasmic reticulum
ET	Electron tomography
FIBSEM	Focused ion beam scanning electron microscopy
FDR	False discovery rate
FNR	False negative rate
FPR	False positive rate
GFP	Green fluorescent protein
GHE	Global histogram equalization
GRP	Gastrin-releasing polypeptide
GUI	Graphical user interface
HAP1	Huntingtin-associated protein-1
HD	Huntington's disease
ICP	Iterative closest point
IF	Invagination factor
INM	Inner nuclear membrane
ipRGC	Intrinsically photosensitive retinal ganglion cell
IVEM	Intermediate voltage electron microscope
LD	Light:dark
MCC	Matthews' correlation coefficient
MPAS	Multiplane automatic segmentation
MRI	Magnetic resonance imaging
MSI	Morphological skeleton interpolation
NBCR	National Biomedical Computation Resource
NE	Nuclear envelope



OC	Optic chiasm
ONM	Outer nuclear membrane
NPC	Nuclear pore complex
NPV	Negative predictive value
PNG	Portable Network Graphics
RAM	Random access memory
ROC	Receiver operating characteristic
ROI	Region of interest
SBEM	Serial block-face scanning electron microscopy
SCN	Suprachiasmatic nucleus
SEM	Scanning electron microscope
SGE	Sun Grid Engine
SLASH	Scalable system for large data analysis and segmentation utilizing a hybrid approach
SNR	Signal-to-noise ratio
ssET	Serial section electron tomography
ssSEM	Serial section scanning electron microscopy
ssTEM	Serial section transmission electron microscopy
STB	Stigmoid body
SVR	Surface area to volume ratio
TEM	Transmission electron microscopy
TNR	True negative rate
TPR	True positive rate
UTSL	Ultrathin section library
UHVEM	Ultrahigh voltage electron microscope
VIP	Vasoactive intestinal polypeptide
VL	Ventrolateral
WT	Wildtype

## LIST OF FIGURES

Figure 1.1.	A schematic of the SBEM imaging process .....	10
Figure 1.2.	A comparison of the pixel sizes and total image volumes used in a sampling of publications that employed 3D EM methods.....	24
Figure 2.1.	The manual segmentation of organelles from SBEM image stacks represents a significant bottleneck to quantitative analyses .....	33
Figure 2.2.	The collaborative segmentation workflow.....	36
Figure 2.3.	The instructions given to volunteers for the task of segmenting nuclei and nucleoli .....	37
Figure 2.4.	An online portal to facilitate collaborative segmentation .....	38
Figure 2.5.	A discussion board for volunteers to ask and answer questions encountered while segmenting.....	39
Figure 2.6.	The proposed method for image downsampling processes entire datasets without stalling .....	52
Figure 2.7.	The automatic detection of image borders .....	56
Figure 2.8.	A flowchart of the steps involved in training data generation .....	61
Figure 2.9.	ROC curves for CHM classifiers of various organelles .....	65
Figure 2.10.	The binarization of probability maps using active contours initialized by a multi-level Otsu threshold yields accurate segmentation results .....	68
Figure 2.11.	Binarization of probability maps using active contours outperforms other methods.....	71
Figure 2.12.	The results of the proposed method are consistent when applied to diverse organelle targets.....	72
Figure 2.13.	A full-slice CHM probability map for nuclei .....	73
Figure 2.14.	A full-slice ilastik probability map for nuclei.....	75
Figure 2.15.	ROC and precision-recall curves for CHM nuclear classifiers.....	77
Figure 2.16.	ROC and precision-recall curves for an ilastik voxel classifier .....	78
Figure 2.17.	Input images can be downsampled to various degrees before the segmentation results are negatively affected.....	80
Figure 2.18.	Automatic segmentation can be efficiently scaled to handle full slices from teravoxel-sized SBEM datasets.....	81
Figure 3.1.	Nuclear invaginations are specific to certain populations of cells.....	92
Figure 3.2.	The MPAS algorithm produces accurate probability maps for re-sliced data in different orientations using a single pixel classifier.....	99
Figure 3.3.	The application of MPAS to the automatic segmentation of nuclei helps properly classify pixels near the nuclear envelope .....	101
Figure 3.4.	A single interslice interpolation between two differently scaled and translated circles .....	107
Figure 3.5.	Multiple interslice interpolations between two differently scaled and translated circles .....	109

Figure 3.6.	Multiple interslice interpolations between two differently scaled and rotated squares .....	111
Figure 3.7.	Multiple interslice interpolations between two differently scaled and rotated segmentations of an invaginated nucleus.....	113
Figure 3.8.	An example scenario in which automatic segmentation accuracy benefits from post-processing by interslice interpolation.....	115
Figure 3.9.	Replacing poorly segmented slices with interslice interpolations increases morphological accuracy .....	117
Figure 3.10.	A flowchart of the steps involved in contour and mesh generation from large-scale automatic segmentations .....	121
Figure 3.11.	Automatically generated surface renderings of nuclei from the ZT04 SCN SBEM dataset.....	124
Figure 3.12.	Automatically generated surface renderings of nucleoli from the ZT04 SCN SBEM dataset .....	126
Figure 3.13.	Combined renderings of automatically segmented nuclei and nucleoli from the ZT04 SCN SBEM dataset.....	128
Figure 3.14.	An example of the nuclear morphological characterization workflow	132
Figure 3.15.	An example of the nucleolar positioning workflow .....	134
Figure 3.16.	An example of the nucleolar positioning workflow, Continued .....	136
Figure 3.17.	The 3D convex hull for a single nucleus .....	143
Figure 3.18.	3D convex hull renderings for nuclei from a full SBEM volume.....	145
Figure 3.19.	The local shape index scalar field for a nucleus with a single invagination.....	147
Figure 3.20.	The local shape index scalar field for a nucleus with multiple invaginations .....	149
Figure 3.21.	The local shape index scalar field for a heavily invaginated nucleus	151
Figure 3.22.	The shape index and convex hull difference are able to discern small qualitative differences in nuclear invagination .....	153
Figure 3.23.	Neuronal compartments were delineated by manual segmentation of the plasmalemma .....	155
Figure 3.24.	Binary masks generated from neuronal segmentations are used to separate organelles into their proper cellular compartments .....	157
Figure 3.25.	Dataset-wide renderings of mitochondria belonging to three SCN neurons.....	159
Figure 4.1.	An overview of SCN neuroanatomy .....	166
Figure 4.2.	The core transcriptional-translational feedback loop of the mammalian circadian clock .....	170
Figure 4.3.	Stigmoid bodies are cytoplasmic inclusions found in SCN neurons..	190
Figure 4.4.	The distribution of nuclear volumes in SCN neurons .....	199
Figure 4.5.	The distribution of nuclear surface areas in SCN neurons .....	200
Figure 4.6.	The mean values of topological descriptors for nuclear invagination in nuclei of SCN neurons.....	201

Figure 4.7.	Electron tomography of SCN nuclei reveals the ultrastructural characteristics of nuclear invaginations .....	202
Figure 4.8.	The distribution of total nucleolar volume in SCN neurons and the percentage of neurons containing multiple nucleoli .....	204
Figure 4.9.	The distribution of nucleolar volume fraction in SCN neurons .....	206
Figure 4.10.	The distribution of stigmoid body volume in SCN neurons and the percentage of neurons containing stigmoid bodies .....	207
Figure 4.11.	The mean values of topological descriptors for stigmoid bodies in SCN neurons .....	209
Figure 4.12.	Stigmoid bodies contain tunnels associated with the endoplasmic reticulum .....	210
Figure 5.1.	A survey of the SBEM literature revealed that the vast majority of studies did not employ automatic or semi-automatic analyses .....	217
Figure 5.2.	A flowchart of the workflow demonstrating proposed future developments .....	220
Figure 5.3.	A flowchart of the workflow demonstrating proposed future developments that includes automatic corrections .....	222
Figure 5.4.	A demonstration of preliminary results from the automatic workflow for classifying mitochondrial morphology .....	224
Figure 5.5.	A subset of ten manually segmented neurons.....	226

## LIST OF TABLES

Table 2.1.	An expedited approach to the downsampling of SBEM image stacks	51
Table 2.2.	Computational requirements for organelle-specific pixel classification .....	63
Table 2.3.	Segmentation evaluation metrics for the tested organelle targets using various methods of probability map binarization .....	69
Table 3.1.	A nuclear segmentation generated automatically by MPAS yields a more faithful representation of ground truth morphology .....	98
Table 3.2.	The metrics automatically computed and output during the morphological analysis of nuclei .....	131
Table 3.3.	The shape index and convex hull difference are able to discern small qualitative differences in nuclear invagination .....	141
Table 4.1.	A survey of significant chromorphological studies and their pertinent results .....	179
Table 4.2.	Imaging parameters used during SBEM dataset acquisition .....	186
Table 4.3.	Imaging parameters used for the collection of tomographic tilt series of SCN nuclei .....	193
Table 4.4.	The results of nuclear morphological characterization for three SCN SBEM datasets .....	194
Table 4.5.	Nucleolar volume and positioning in SCN neurons with single and multiple nucleoli .....	196

## ACKNOWLEDGEMENTS

As a high school senior writing my application to UCSD, I remember looking down the list of majors offered, seeing bioengineering, and thinking to myself “what is that?” Quite frankly, I had no clue. But I knew that I liked biology, computers, and playing with numbers, so maybe this was the field for me. I applied to the major on a whim, got accepted, stumbled around for my first year, fell in love with science during my second, and wound up staying for a Ph.D. Despite so much uncertainty, there has always been one constant throughout my time at UCSD – the support of my family. From my decision to move to San Diego to my decision to enter graduate school, I have always had the unequivocal backing and guidance of my parents, and for that I will always be grateful.

Throughout the years, my passion for microscopy has truly been cultivated by my mentors at NCMIR. First and foremost, I’d like to thank my advisor, Mark Ellisman, for affording me the opportunity to spend my pre-doctoral years working at the cutting edge of the field. It was through his supervision that I was encouraged to collect more data than I knew what to do with while being provided with the tools and support necessary to engineer innovative ways to analyze it. My undergraduate advisor, Guy Perkins, has also played a huge role in the development of my scientific career. Rather than view me in the manner by which many scientists look at undergraduates, he trained me like a true researcher and gave me independent projects to work on that directly inspired me to pursue a Ph.D. I will always be thankful to him for this approach to mentorship.

My dissertation committee members have provided valuable and encouraging feedback throughout the formulation of the work presented in the following pages. I am thankful for the guidance they have provided in helping this effort come to fruition. In particular, I would like to thank Gina Sosinsky for kindly taking the time to discuss my

project and help brainstorm the most logical way to tell this story. For graciously providing the funding that made the bulk of my graduate studies possible, I would like to thank the San Diego Fellowship organization and Vivian Hook for granting me a position on her NIH/NIDA training grant.

One of the greatest perks of working in a lab as diverse as NCMIR has been the ability to interact with and learn from a talented and eclectic group of neuroscientists, engineers, and computer scientists, both on-campus and off. Much of the work presented in this dissertation has been performed in collaboration with the labs of Satchin Panda at the Salk Institute and Tolga Tasdizen at the SCI Institute of the University of Utah, and I will always be indebted to them for their encouraging feedback and the myriad opportunities they have provided for me. I would like to particularly thank Mojtaba Seyedhosseini from SCI for introducing me to the cascaded hierarchical model and teaching me how to apply it to biological data. The animals used in the studies presented here were provided by Satchin Panda's lab, and tissues, grids, and datasets were prepared and acquired in collaboration with Tom Deerinck, Monica Berlanga, Eric Bushong, Christine Kim, and Andrea Thor at NCMIR. I am thankful to them for providing the animals and much of the data that made this dissertation possible. I am especially thankful to Monica Berlanga, whose mentorship and guidance were greatly influential throughout my time at NCMIR. Much of the work presented here was greatly inspired by our work together, and I will always be appreciative of her for truly making me feel like an integral part of the lab for the first time.

In addition to these collaborations, I have had the unique privilege of being trained, mentored, and assisted by so many other wonderful people at NCMIR that it seems foolish to list them all, but I'm going to try anyways. David Lee, Chris Churas, and Willy Wong provided me with much support and guidance on the computational side of my project,

and many of the large-scale segmentations presented here would not have been possible without their help and access to computational resources and seemingly endless supplies of CPU years. I am very thankful for the great EM training provided to me by Mason Mackey, as well as the many laughs and trips to La Costa. James Bower, Tristan Shone, and Tomas Molina provided invaluable guidance and troubleshooting during the collection and processing of the bulk of the tomographic data presented in this thesis. Much of my desire to pursue cellular reconstructions at large scales was inspired by discussions with Andrew Noske and Rick Giuly, and I am appreciative of their time and assistance. Edmond Negado, Vicky Rowley, and the rest of the IT team at NCMIR provided me with great support and put up with accommodating and moving around massive amounts of my data. I promise I'll clean it up soon.

Last but not least, I would like to thank the many people who have made my time as a graduate student at NCMIR enjoyable on a social level. Tapi, Nate, Don, Felix, Niko, Phuong, Guillaume, Mason, Andrea, James, Tristan, Monica, Andrew, Rick, and so many others. From lunches and birthday parties to celebrating the Giants' World Series victories to the NCMIR fantasy football league and World Cup pool. Being in the same lab as all of you has truly been a blast. Thank you for the great memories.

Chapter 2, in part, is a reprint of the material as it appears in *Frontiers in Neuroanatomy*, 2014, 8(126). Perez, A.J., Seyedhosseini, M., Deerinck, T.J., Bushong, E.A., Panda, S., Tasdizen, T., and Ellisman, M.H. The dissertation author was the primary investigator and author of this paper.



## VITA

- 2004-2005 Scientific Intern, Department of Vascular Biology, The Scripps Research Institute, La Jolla, California
- 2006-2007 Research Assistant, National Center for Microscopy and Imaging Research, University of California, San Diego
- 2007 Bachelor of Science, Department of Bioengineering, University of California, San Diego
- 2007 Teaching Assistant, Department of Mechanical Engineering, University of California, San Diego
- 2008 Master of Science, Department of Bioengineering, University of California, San Diego
- 2008-2010 Teaching Assistant, Department of Bioengineering, University of California, San Diego
- 2009-2014 Graduate Student Researcher, National Center for Microscopy and Imaging Research, University of California, San Diego
- 2014 Doctor of Philosophy, Department of Bioengineering, University of California, San Diego

## PUBLICATIONS

Perez, A.J., Seyedhosseini, M., Deerinck, T.J., Bushong, E.A., Panda, S., Tasdizen, T., and Ellisman, M.H. (2014). A workflow for the automatic segmentation of organelles in electron microscopy image stacks. *Frontiers in Neuroanatomy*, 8(126), 1-13. doi:10.3389/fnana.2014.00126

Perkins, G.A., Scott, R., Perez, A., Ellisman, M.H., Johnson, J.E., and Fox, D.A. (2012). Bcl-xL-mediated remodeling of rod and cone synaptic mitochondria after postnatal lead exposure: Electron microscopy, tomography, and oxygen consumption. *Molecular Vision*, 18, 3029-3048. PMID: 23288995.

Kim, C.E., Perez, A., Perkins, G.A., Ellisman, M.H., and Dauer, W.T. (2010). A molecular mechanism underlying the neural-specific defect in torsinA mutant mice. *Proceedings of the National Academy of Sciences U.S.A.*, 107(21), 9861-9866. doi: 10.1073/pnas.0912877107

Perkins, G.A., Sosinsky, G.E., Ghassemzadeh, S., Perez, A., Jones, Y., and Ellisman, M.H. (2008). Analysis of the cross-bridges in the paranodal region of the node of Ranvier by electron tomography. *The Journal of Structural Biology*, 161, 469-480. doi:10.1016/j.jsb.2007.10.005

Saini, V., Martyshkin, D.V., Mirov, S.B., Perez, A., Perkins, G., Ellisman, M.H., Wu, H., Pereboeva, L., Borovjagin, A., Curiel, D.T., and Everts, M. (2008). An adenoviral platform for selective self-assembly and targeted delivery of nanoparticles. *Small*, 4(2), 262-269. doi: 10.1002/sml.200700403

Saini, V., Enervold, M.R., Perez, A., Koploy, A., Perkins, G., Ellisman, M.H., Green, H.N., Mirov, S.B., Zharov, V.P., and Everts, M. (2007). "Targeting nanoparticles to tumors using adenoviral vectors," in *2007 NSTI Nanotechnology Conference*, 2, 321-324.

## ABSTRACTS & CONFERENCE PROCEEDINGS

Perez, A.J., Seyedhosseini, M., Churas, C., Kim, K.-Y., Hatori, M., Bushong, E.A., Deerinck, T.J., Le, H., Panda, S., Tasdizen, T., and Ellisman, M.H. (2014). Workflows for the automatic segmentation and characterization of organelle morphology and distribution in electron microscopy image stacks. *Society for Neuroscience 2014*, Washington, D.C., U.S.A.

Perez, A.J., Seyedhosseini, M., Tasdizen, T., Ellisman, M.H. (2014). Automated workflows for the morphological characterization of organelles in electron microscopy image stacks. *Experimental Biology 2014*, San Diego, CA, U.S.A.

Perez, A.J., Berlanga, M.L., Kim, K.-Y., Johnson, D.D., Hatori, M., Bushong, E.A., Deerinck, T.J., Le, H., Seyedhosseini, M., Giuly, R.J., Lee, D., Jurrus, E., Tasdizen, T., Ellisman, M.H., Panda, S. (2013). Morphological plasticity of the mouse suprachiasmatic nucleus revealed by a multiscale imaging approach. *Society for Neuroscience 2013*, San Diego, CA, U.S.A.

Schachtrup, C., Ryu, J.K., Carlton, P.M., Le Moan, N., Perez, A., Vagena, E., Ellisman, M.H., Wyss-Coray, T., Akassoglou, K. (2013). Regulation of astrocyte activation by the cleaved p75 neurotrophin receptor. *XI European Meeting on Glial Cell Function in Health and Disease*, Berlin, Germany.

Schachtrup, C., Ryu, J.K., Carlton, P.M., Le Moan, N., Perez, A., Ellisman, M.H., Wyss-Coray, T., Akassoglou, K. (2012). Regulation of astrocyte activation by the cleaved p75 neurotrophin receptor. *NGF 2012*, Würzburg, Germany.

Fox, D.A., Perkins, G.A., Johnson, J.E., Giddabasappa, A., Chaney, S., Brown, J., Lahsaei, P., Ghassemzadeh, S., Perez, A., Dixit, A., Ellisman, M.H. (2007). Differential susceptibility of rod photoreceptor synaptic and non-synaptic mitochondria to divalent cation exposure: Neuroprotection by Bcl-xL overexpression. *Mitochondrial Medicine*, Pacific Beach, CA, U.S.A.

Fox, D.A., Perkins, G.A., Johnson, J.E., Giddabasappa, A., Chaney, S., Brown, J., Lahsaei, P., Ghassemzadeh, S., Perez, A., Dixit, A., Ellisman, M.H. (2007). Differential susceptibility of rod photoreceptor synaptic and non-synaptic mitochondria (Mt) to lead and protection by Bcl-XI. *ARVO Meeting*, Fort Lauderdale, FL, U.S.A.

Saini, V., Enervold, M.R., Perez, A., Koploy, A., Perkins, G., Ellisman, M.H., Green, H.N., Mirov, S.B., Zharov, V.P., Everts, M. (2007). Targeting nanoparticles to tumors using adenoviral vectors. *NSTI Nanotechnology (BioNano) Conference*, Santa Clara, CA, U.S.A.

Saini, V., Perez, A., Koploy, A., Perkins, G., Ellisman, M.H., Nikles, D.E., Everts, M. (2007). Adenoviral platform for selective assembly and targeted delivery of gold nanoparticles to tumor cells. *Nanotechnology in Biomedicine, Keystones Symposia*, Tahoe City, CA, U.S.A.

Fox, D.A., Perkins, G.A., Johnson, J.E., Giddabasappa, A., Chaney, S., Brown, J.M., Wu, I., Lahsaei, P., Ghassemzadeh, S., Perez, A., Dixit, A. and Ellisman, M.H. (2006). Differential regulation and susceptibility of mouse rod photoreceptor inner segment and synaptic terminal mitochondria to divalent cation-induced overload. *XVII International Congress of Eye Research*, Buenos Aires, Argentina.

ABSTRACT OF THE DISSERTATION

The Automated Reconstruction and Analysis of High Resolution Spatial Models of  
Neuronal Microanatomy

by

Alexander Joseph Perez

Doctor of Philosophy in Bioengineering

University of California, San Diego, 2014

Professor Mark H. Ellisman, Chair

Professor Gabriel A. Silva, Co-Chair

Electron microscopy (EM) facilitates analysis of the structure, distribution, and functional status of organelle networks within the nervous system. Recent breakthroughs in EM specimen preparation and instrumentation have furnished scientists with the ability to automatically collect volumetric datasets large enough to cover significant swaths of

neuroanatomical subdivisions at nano-resolution. The quantification of biological morphologies from these data, however, typically requires image segmentation, which is a long-standing and well-recognized bottleneck. Though datasets may now be collected at rates exceeding teravoxels per day, the manual segmentation and analysis of all features from such a volume requires many years of human labor. As technological advances driven by the desire to reconstruct entire nervous systems continue to push instrument throughput skyward, it is clear that our ability to model brain ultrastructure will be limited by the rate of image analysis rather than that of image acquisition.

The body of work described in this dissertation represents a contribution towards alleviating this impediment. A pipeline for the automatic segmentation, morphological quantification, and spatial characterization of organelles from high resolution datasets at the teravoxel-scale is presented. Segmentations were generated using a highly parallelized, supervised machine learning approach that reduces the required human effort from years to just a few hours. A host of generic and organelle-specific post-segmentation filters were developed, and it is shown that their application improves segmentation accuracy. Accelerated approaches for generating surface renderings from these large-scale segmentations are introduced, and a workflow for the automatic computation and reporting of morphological, topological, and spatial metrics is described. These methods were then applied to study the spatiotemporal changes of organelles in neurons of the mouse suprachiasmatic nucleus across the diurnal cycle. Novel findings pertaining to nuclear structure and organization are reported and discussed. Taken together, the methods described here provide a series of tools for expediting the quantitative analysis of organelle structure-function relationships in the current era of big data in biological microscopy.

## **Chapter 1**

# **Methods for Whole Cell Imaging at High Resolution and Their Applications in the Neurosciences**

## **1.1. Introduction**

Over the past decade, the field of neuroscience has experienced a renewed enthusiasm and commitment towards exploring and understanding the structural underpinnings of how the brain works. Such pursuits have generated a palpable buzz in the scientific community that has noticeably extended to the general population. Indeed, the launches of the BRAIN Initiative and Human Brain Project, two ventures driven by the desire to better map and understand the human brain, have brought mainstream attention to a field that was previously bereft of it (Abbott, 2013; Insel et al., 2013). It is widely acknowledged that this rejuvenation is the product of the rapid development and proliferation of technologies for preparing, imaging, and reconstructing regions of the brain at unprecedented scale and resolution (Knott and Genoud, 2013; Peddie and Collinson, 2014). As a result of this technological progress, neuroscientists now have a toolbox of modalities at their disposal that enables the rapid and automatic imaging of large volumes of the brain at the level of ultrastructural, and sometimes molecular, resolution.

The rest of this chapter will serve as an introduction to the individual components of this imaging toolbox, describing their histories, applications, and associated technological breakthroughs. Particular focus will be paid to the ability of each modality to image organelles and other nanoscale features within the subcellular compartment. Finally, the conclusion of this chapter will feature a look towards the future of the field and establish the importance of the technologies developed in this dissertation.

## **1.2. Light microscopy**

From the pioneering neuroanatomical studies of Santiago Ramón y Cajal and Camillo Golgi (Cajal, 1906; Golgi, 1906) to the current wealth of modalities designed to surpass the resolution barrier imposed by light's diffraction limit (Patterson et al., 2010),

light microscopy (LM) and its associated technologies have proven indispensable to the neurosciences. In early studies, such as those of Ramón y Cajal and Golgi, cells could only be visualized if they were first darkly stained to provide contrast in the optical microscope. As an additional hurdle, useful depictions were only possible if this staining was unique to just a very small subset of neurons within the field of view; if too many neurons were dark, the resultant microscopic image would resemble an indecipherable mass of stain. Though unproven at the time, this additional layer of complexity was present because individual fibers of the neuropil are frequently smaller than the wavelength of light, rendering them unresolvable from one another by conventional LM (Denk and Horstmann, 2004). Fortunately, the staining method developed by and named after Golgi achieved this requisite selective labeling via the deposition of silver chromate at neuronal membranes following fixation of tissues with potassium dichromate and silver nitrate (Pannese, 1999). By a mechanism that remains unknown, Golgi's method specifically and randomly labels only a small subset of neurons in their entirety, a fact that made it ideal for early studies on neuronal morphology.

Though such methods laid the groundwork for modern neuroscience, they were limited to the depiction of gross cellular morphologies; intricate views of subcellular compartments remained largely beyond the capabilities of LM alone. This changed with the advent and widespread adoption of fluorescence microscopy, which, combined with significant advances in instrumentation, has enabled neuroscientists to view subcellular components with increasing levels of clarity (Wilt et al., 2009). Though early applications of fluorescence microscopy were limited to organic dyes attached to proteins of interest via antibodies, fluorophores that could directly recognize organelles (Buckman et al., 2001), DNA (Kapuscinski, 1995; Smith et al., 2000), lipids (Gan et al., 2000), and ions (Grynkiewicz et al., 1985; Miyawaki et al., 1997) were subsequently developed. The



introduction of genetically encodable fluorescent proteins, such as green fluorescent protein (GFP), allowed for precise fluorophore targeting via covalent linkage to the protein of interest and the generation of fluorescence without the need for additional cofactors (Tsien, 1998; Giepmans et al., 2006). From an instrumentation standpoint, the commercialization of confocal microscopes (White and Amos, 1987) and the subsequent introduction of two-photon systems (Denk et al., 1990) have enabled researchers to image fluorescent signals with improved clarity from increasing tissue depths. Modalities that allow for the localization of fluorophores at resolutions finer than the diffraction limit of light (Gustafsson, 2000; Betzig et al., 2006; Rust et al., 2006) have found numerous applications in the neurosciences, including the detailed localization of synaptic cytoskeletal filaments (Pielage et al., 2008) and receptor proteins (Dani et al., 2011).

Though the aforementioned breakthroughs have increased our ability to resolve the location of small intercellular components using LM, such visualizations remain restricted to compartments or proteins that have been fluorescently tagged. Structures that have not been tagged remain hidden, and even tagged structures do not yield a continuous view of ultrastructure or membrane topology. Ideally, images of the same region could be acquired using both EM and fluorescence LM and combined with one another to simultaneously yield both molecular localization and fine ultrastructure. This process, known as correlated light and electron microscopy (CLEM), is currently a major focus within the community. One early success in the field of CLEM was the use of quantum dots, which can be engineered to label desired protein targets for LM and possess a molecular weight great enough to scatter electrons and appear opaque in electron micrographs (Giepmans et al., 2005; Silva, 2006). Though such approaches were promising, microscopists, inspired by the success of GFP, desired a probe for CLEM that could be genetically encoded. One such approach involves the use of a genetically

encodable fluorophore that can also generate the singlet oxygen required to polymerize diaminobenzadine (DAB) through a process known as photoconversion (Deerinck et al., 1994). Such DAB precipitates are osmiophilic, and therefore render the same tagged structures visible as electron dense clouds in EM micrographs. This principle led to the development of a number of genetically encodable tags for CLEM, including FLAsH and ReAsH (Gaietta et al., 2002; Sosinsky et al., 2003), miniSOG (Shu et al., 2011), and APEX (Martell et al., 2012). In addition, a recent report has demonstrated the retention of GFP fluorescence following resin embedding and EM preparation (Peddie et al., 2014), a process that provides an alternative, but still genetically encodable, pathway for CLEM.

Although such CLEM techniques are promising, they remain in the early stages of development. Furthermore, though some researchers have acquired large-scale volumes of the brain using multiphoton fluorescence microscopy with stage mosaicking (Chow et al., 2006; Berlanga et al., 2011), Brainbow labeling (Cai et al., 2013), and CLARITY (Chung and Deisseroth, 2013), the resolution of such methods is limited. For example, the volume of  $\alpha$ -synuclein immunoreactivity in the mouse brain acquired by Price and colleagues has a lateral pixel size of 0.24  $\mu\text{m}$  (Price et al., 2006), a value that is far too coarse to resolve the ultrastructure of individual organelles or track membrane curvature. As such, the electron microscope and its related technologies remain uniquely adapted for providing images that can be used to simultaneously study subcellular ultrastructure as well as the connectivity between cells of the nervous system.

### **1.3. Serial section transmission electron microscopy**

The invention of the first glass knife microtome capable of cutting thin sections from plastic-embedded specimens (Porter and Blum, 1953) allowed for TEM-based ultrastructural studies of a number of organelles, including mitochondria (Palade, 1952),

ribosomes (Palay and Palade, 1955), and the endoplasmic reticulum (Palade and Porter, 1954). These ground-breaking studies established much of our baseline knowledge of the structure and function of the machines that drive biological processes at the cellular level. However, since many subcellular components and organelles are significantly larger than the thickness (~50-100 nm) of the sections used for conventional TEM, individual micrographs can be misleading with respect to organelle morphology. The most intuitive first approach to circumventing this problem is to cut thicker sections that have a greater probability of containing entire organelles. Unfortunately, as section thickness increases, so do electron scattering events and chromatic aberration, effects that quickly degrade image quality. Since early TEMs did not operate at voltages sufficient enough to limit these effects by increasing the initial acceleration of the electron beam, microscopists had to develop other methods to explore complete 3D morphologies.

In the first of these methods, known as serial section transmission electron microscopy (ssTEM), ribbons of consecutive thin sections are cut from the block-face using a microtome and collected, in the same order in which they were cut, onto EM grids (Gay and Anderson, 1954; Sjöstrand, 1958). The same region of interest (ROI) is then imaged from each section, resulting in a stack of images spaced apart by the cutting thickness of the microtome. Such a stack can then be used to track individual organelles or neuronal processes across sections, producing complete and high resolution 3D morphologies. The development and evolution of methods to furnish such 3D reconstructions are discussed in detail in Chapter 2.

Early studies using ssTEM explored the frog muscle spindle (Karlsson et al., 1966) and studied the organization of organelles in neuronal somata of the rat lateral geniculate nucleus (Karlsson, 1966). In the latter study, quantitative data, including length, surface area, and volume, of organelles such as the Golgi apparatus, mitochondria, and nucleus

were provided. A sampling of subsequent studies reveals that ssTEM has been employed to establish both structure (Fiala et al., 1998; Harris, 1999; Huang et al., 1998) and connectivity (Sjöstrand, 1974; Chalfie et al., 1985; Hall and Russell, 1991; Mishchenko et al., 2010; Cardona et al., 2010) in the brain. In a seminal study conducted by a team led by Sydney Brenner at the MRC Laboratory for Molecular Biology, the entire nervous system of the nematode *Caenorhabditis elegans*, including all neuronal processes and synapses, was mapped using ssTEM (White et al, 1986). Though this task was certainly simplified by the fact that the entire *C. elegans* nervous system contains only 302 neurons, this study remains the only instance in which the entire neuronal wiring diagram, or connectome, of any organism has been successfully mapped.

Despite the fact that ssTEM still enjoys widespread use (Lu et al., 2014; Fuchs et al., 2014), its labor-intensive reputation is well established. Even today, all steps involved in the process, including specimen preparation, section cutting and collecting, imaging, and reconstruction, require some degree of interaction by highly trained experts. Consequently, the technique is highly prone to human error; if sections or images from the middle of a series are lost or damaged, the whole series may be jeopardized. As a result of such errors, the reconstruction of the *C. elegans* connectome necessitated the combination of images from different regions of several worms (Seung, 2013). Further, even if the high risk for human error is ignored, the sheer task of collecting large volumes with ssTEM remains daunting. The *C. elegans* dataset consisted of images from roughly 8,000 sections cut to thicknesses of 50 nm (White et al., 1986), and Karlsson's datasets of the frog muscle spindle approached 10,000 sections each (Karlsson et al., 1996). On account of these astonishingly large numbers, it is readily apparent that ssTEM would become much more feasible if the need to manually cut and collect sections were removed from the equation. Moreover, one can imagine that further ease would be introduced if the

imaging process could be automatically synchronized to coincide with each successive cut. Fortunately, practitioners of the field were in luck; the introduction of the serial block-face scanning electron microscope (SBEM; Leighton, 1981; Denk and Horstmann, 2004) simultaneously achieved both of these goals and revolutionized the field of large-scale 3D EM.

#### **1.4. Serial block-face scanning electron microscopy**

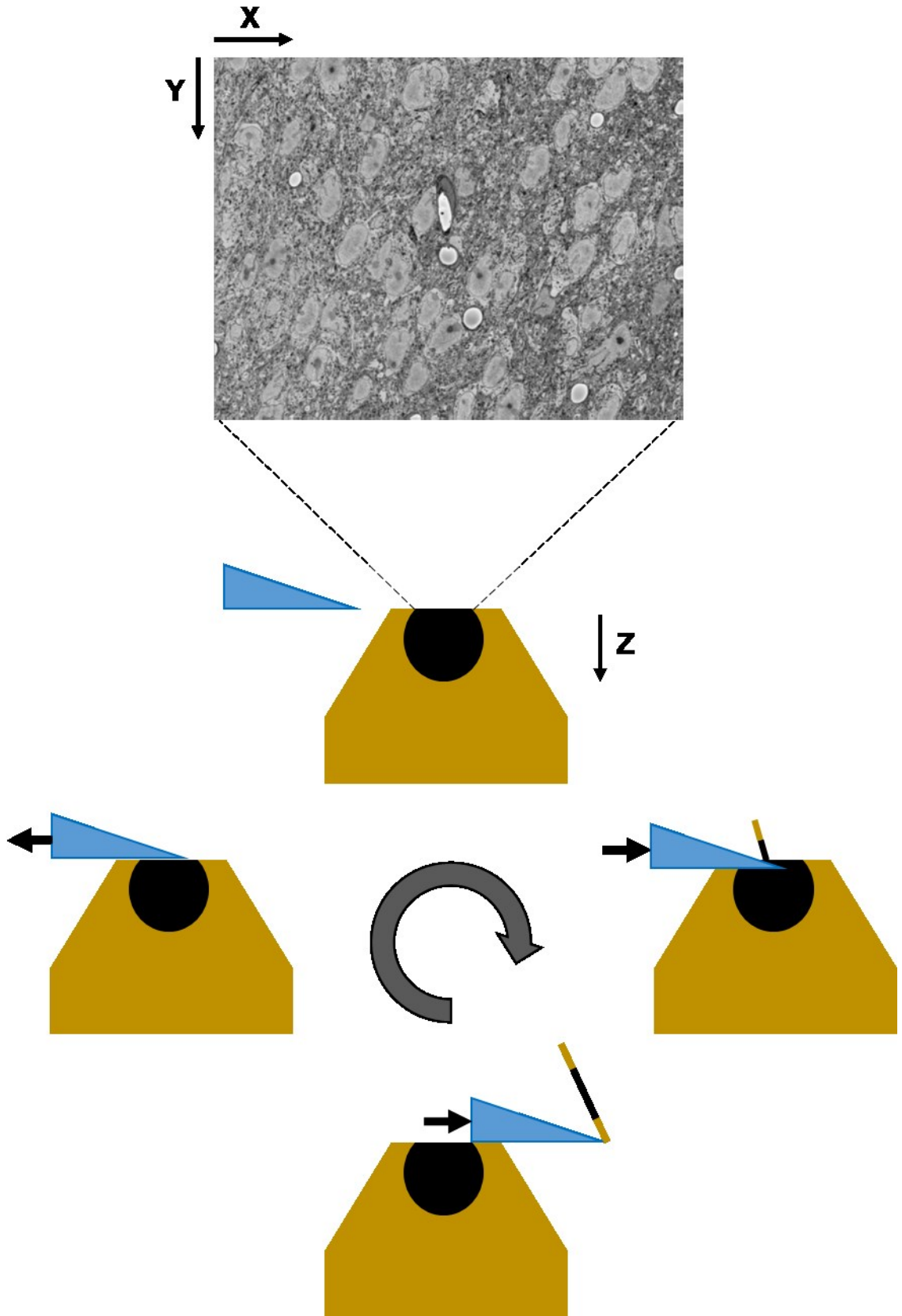
With the aid of hindsight, the idea that led to the invention of SBEM seems relatively intuitive: instead of cutting thin sections from a block and producing images of these sequential sections, the block itself is imaged following repeated section removals. If this continuously repeating cycle of cuts followed by image acquisitions could be automated, the entire process of ssTEM would be emulated without the need for human manipulation, thereby significantly reducing the risk of section loss or damage (Figure 1.1). Furthermore, if the surface of the block were imaged at a fixed position relative to the primary electron beam and detector, the output stack of images would already be essentially aligned from one section to the next without the risk of section warping or the need to re-align the microscope following each cut.

A machine capable of automating this process requires two principle components: (1) a microscope other than the TEM, whose electron beam must penetrate the sample to produce image contrast, and (2) a means to automatically plane thin sections off of the block-face from within the chamber of the microscope. Both of these needs were first addressed by Leighton (Leighton, 1981) who, in 1981, developed an ultramicrotome capable of cutting sections off of a resin-embedded block from within the chamber of a scanning electron microscope (SEM). The SEM used by Leighton, which produced image contrast based on the detection of electrons emitted from the sample's surface when it

was scanned by a primary electron beam, seemed perfect for imaging block-faces. However, Leighton's images based on the detection of these so-called secondary electrons were marred by a significant artifact: surface charging. Plastic-embedded biological samples are not naturally conductive and therefore act as insulators in the SEM, trapping electrons at and just below the surface of the sample. This accumulation of negative charge leads to a surface potential across the block face and a deceleration of incident electrons. Due to the heterogeneous nature of biological tissues, these reduced landing energies are non-uniform across the block, a problem that causes significant distortions in the resultant image. For this reason, Leighton had to remove the sample from the chamber and coat it with a layer of conductive metal before satisfactory images could be obtained. Since such coating steps made full automation impossible, and because image collection and storage systems remained primitive in 1981, Leighton's invention did not immediately catch on.

One approach to mitigating the impact of surface charging when imaging non-conductive specimens is the use of an environmental scanning electron microscope (ESEM), which maintains a low concentration of gas within its chamber (Donald, 2003). Positively charged ions are generated as the primary electron beam impinges upon gas molecules in the ESEM's chamber, and these ions serve to neutralize the negative charge that accumulates at the block-face. The use of this "low-vacuum" mode of operation was the first of many innovations employed by Denk in his version of the SBEM (Denk and Horstmann, 2004). Additionally, Denk opted to use a significantly higher beam accelerating voltage (7.5 keV) to allow for the detection of backscattered electrons (BSEs), which are incident electrons from the primary beam that have been elastically scattered out of the specimen's interacting volume due to collisions with its atoms. Importantly, BSE

**Figure 1.1. A schematic of the SBEM imaging process.** (Top) A BSE image of the block-face (yellow) is acquired while the diamond knife (blue) is retracted. After this image has been acquired, the block is advanced in the direction of the knife by the desired axial step size, a value that typically falls in the range of 20-100 nm. The diamond knife is then advanced across the block-face (right), planing off a section of the user-specified thickness (bottom). These loose sections can often accumulate on the knife, and therefore must be removed from time to time to minimize the risk of block-face occlusion. The knife is then retracted across the planed block-face to its initial position (left). A new image is acquired once it has been fully retracted, thus starting the cycle anew. The X and Y dimensions are specified by the raster size at the plane of the block-face, while the Z dimension is specified by incremental steps through the depth of the block. The black semi-circle represents tissue that has been processed for SBEM imaging to provide optimal contrast and surface conductivity.





detection holds a major advantage over secondary electron detection when imaging the block face: BSE scattering is strongly dependent upon the charge of the atomic nucleus that the primary electron collides with. This creates ideal contrast when imaging blocks embedded with conventional heavy metal stains, and the BSE images acquired by Denk closely resembled conventional TEM micrographs. To maintain the block in the same position for optimal slice-to-slice registration, Denk designed a custom diamond knife microtome in which the block is advanced by a specified amount prior to cutting. Such a design maintains the lateral position of the block-face as a constant and facilitates registration without the need to re-focus after each cut (Denk and Horstmann, 2004).

Unfortunately, a major disadvantage of imaging in low-vacuum mode lies in the fact that the gas molecules within the chamber can scatter primary and backscattered signals, leading to a reduction in signal-to-noise ratio (SNR). Therefore, the dwell time, measured as the time the primary electron beam must spend to generate one pixel on the detector, must be increased to provide more signal. Although dwell times used for SBEM are typically no more than a few microseconds, the size of current detectors, such as the one used in this dissertation ( $7.68 \times 10^8$  pixels), is large enough to make this the rate-limiting step of the SBEM process. Ideally, the SEM chamber should be maintained at as high of a vacuum as possible to allow for decreased dwell times. However, this would of course require an alternate method for increasing specimen conductivity. One obvious way to achieve this is to coat the block face with a thin layer of metal after each cut has been made. Though a device capable of this in-chamber coating has been produced (Titze and Denk, 2013), it is still in the early stages of development. Furthermore, such layers may also decrease SNR by generating BSEs of their own that do not contain information about the block-face. An alternative method introduced by Deerinck and colleagues, and one that has proven very successful, is to enhance conductivity by increasing the amount

of heavy metals deposited while the tissue is being stained (Deerinck et al., 2010). This is done using an osmium tetroxide-thiocarbohydrazide-osmium tetroxide (OTO) stain to increase osmium deposition followed by uranyl acetate treatment and *en bloc* lead aspartate staining. Specimens stained in such a manner produce high contrast images mostly devoid of surface charging artifacts even in chambers held at ultrahigh vacuum levels, thereby allowing pixel dwell times to be decreased.

The automated nature of SBEM means that the amount of data that can be collected from one block is practically bounded by how long the experimenter wants to leave the machine running and how much storage space they have available. A single-slice, 16-bit SBEM image of pixel dimensions 32,000 x 24,000 requires approximately 1.54 GB of hard disk space. This means that typical image stacks in the range of 1,000 sections require multiple terabytes of space just to store the raw images. Further, as a conservative estimate, the storage space needed should be doubled to account for any ensuing post-processing or reconstruction steps. In some of the largest scale SBEM studies to be published thus far, researchers working with Denk used mosaicking to collect datasets with dimensions of 8,192 x 7,072 x 3,200 ( $1.2 \times 10^6 \mu\text{m}^3$ ; Helmstaedter et al., 2013) and 3,584 x 21,658 x 13,000 ( $6.3 \times 10^6 \mu\text{m}^3$ ; Briggman et al., 2011) voxels. These datasets were used to create connectomics-based wiring diagrams of circuits in the mouse retina. Although most large-scale SBEM reconstructions have been inspired by connectomics, a number of smaller scale studies have dealt with more manageable biological questions, including the organization of chromatin (Rouquette et al., 2009), the volumes of synaptic boutons and dendritic spines (Wilke et al., 2014), and the lengths of collagen fibrils (Kalsou et al., 2013).

Image stacks generated by SBEM can cover a wide range of pixel and raster sizes, making the modality adaptable to the various goals of individual experiments. A recent

survey of the SBEM literature found that reported lateral resolutions ranged from 5-80 nm/pixel, while reported axial resolutions ranged from 25-100 nm/slice (Peddie and Collinson, 2014). One commonality is that in almost all practical use cases, SBEM is an anisotropic imaging modality; the cutting thickness tends to be many times greater than the lateral pixel size. While this is typically not an issue, some segmentation and reconstruction algorithms yield superior results when voxel dimensions are isotropic (Sommer et al., 2011). Additionally, some features, such as ER sheets, synaptic vesicles, and postsynaptic densities, are difficult to reliably track across sections spaced >20 nm apart. For experiments in which a finer axial resolution is desired, another serial block-face imaging technique, known as focused ion beam scanning electron microscopy (FIBSEM), was developed. FIBSEM, like SBEM, allows for the acquisition of serial BSE images of the block-face following section removal, but achieves this removal by a different mechanism – the ablation of material from the block’s surface using a focused beam of gallium ions.

### **1.5. Focused ion beam scanning electron microscopy**

FIBSEM systems consist of a dual-beam microscope with both a scanning electron beam used for imaging and a focused beam of gallium ions used for surface ablation (Heymann et al., 2006; Knott, G. et al., 2008). Samples for FIBSEM imaging can be prepared using the same staining and embedding procedures employed for ssTEM and SBEM (Bushby et al., 2011), and images with contrast and lateral resolution comparable to those attainable by SBEM are achieved by the detection of BSEs under high vacuum using electron beam voltages in the range of 2-5 kV (Knott, G. et al., 2008). As previously discussed, the main advantage of FIBSEM over SBEM is its improved axial resolution; the use of a focused ion beam allows for the milling of sections from the block-face with

thicknesses as small as 3-15 nm (Peddie and Collinson, 2014). As such, FIBSEM datasets can achieve isotropic resolutions comparable to the lateral resolution attainable by TEM. Another advantage of FIBSEM lies in the fact that only the portion of the block-face targeted for ablation by the focused ion beam is irreversibly destroyed. This affords the microscopist the ability to collect a dataset from a small section of the block, then re-sample other regions of interest at a later time. This is not possible when using the diamond knife sectioning employed by SBEM, which irreversibly removes entire sections of the block-face.

Unfortunately, the increased axial resolution afforded by FIBSEM comes at the expense of data collection speed and the attainable field of view. A recent report has stated that consistent ion beam milling with 4-5 nm/pixel resolution can only be achieved over, at most, a 20  $\mu\text{m}$  x 20  $\mu\text{m}$  region of the block-face (Knott and Genoud, 2013). Though such a size may be sufficient for reconstructing patches of neuropil, it is only large enough to contain perhaps a few neuronal somata. This limitation, combined with the fact that ion beam milling is significantly slower than diamond knife sectioning, has drastically restricted the size of FIBSEM datasets. As a consequence, the majority of studies employing FIBSEM have examined relatively small volumes in the range of 10 - 3,000  $\mu\text{m}^3$  (Peddie and Collinson, 2014). An example of one such study was that of Wei and colleagues, who used FIBSEM to reconstruct the organelle networks of high-pressure frozen, freeze substituted *Saccharomyces cerevisiae* cells (Wei et al., 2012). Their reconstruction was generated from a FIBSEM dataset with 3 nm isotropic voxels and a total volume of approximately 8  $\mu\text{m}$  x 10  $\mu\text{m}$  x 8  $\mu\text{m}$  (640  $\mu\text{m}^3$ ). The collection of this dataset took about 35 hours, which is a rather substantial period of time for such a small volume.

Despite these limitations, some relatively large volumes have been collected using FIBSEM (70,000  $\mu\text{m}^3$ : Bushby et al., 2011; 1.7x10<sup>6</sup>  $\mu\text{m}^3$ : Armer et al., 2009). Such efforts,

however, are currently the exception rather than the norm. Though the improved axial resolution afforded by FIBSEM makes it preferable for the study of small features, such fine resolution is often not necessary for larger structures, such as membrane-bound organelles. A single mitochondrion is likely to persist across 10s to 100s of axial steps on the order of 30-60 nm, and a nucleus will persist across many 100s of such steps. Even in cases where mitochondria are highly branched, individual branches can be reliably tracked across thicker SBEM slices without issue. Taking this, as well as the desire to rapidly collect larger volumes, into consideration, SBEM was chosen as the primary imaging modality for this dissertation. Each SBEM dataset used in this dissertation has an axial resolution of 30 nm, covers roughly  $600,000 \mu\text{m}^3$ , and was collected in about 5-6 days. This represents an almost 1,000-fold increase in tissue volume with only a 4-fold increase in collection time over the FIBSEM dataset recently reported by Wei and colleagues (Wei et al., 2012).

In recent years, electron tomography (ET), another high resolution, isotropic, 3D EM technique, has been increasingly applied as a complementary modality for SBEM-centric studies (West et al., 2010; Boassa et al., 2013; Kalson et al., 2013; Vihinen et al., 2013; Wong et al., 2013). While SBEM datasets achieve large fields of view at coarser resolutions, ET, similarly to FIBSEM, achieves finer, isotropic voxel dimensions over smaller fields of view. Unlike FIBSEM, however, sequential slices through ET datasets are generated by virtual, *in silico* reconstruction rather than physical sectioning. Additionally, ET reconstructions typically feature equal or finer resolutions when compared to FIBSEM image stacks (Gan and Jensen, 2012), and their acquisition takes on the order of one to two hours rather than days. In the following subsection, the history and theory of ET will be reviewed. This will be followed by a statement of ET's power as a complementary technique for serial block-face modalities and its applicability to the present study.

## 1.6. Electron tomography

Early biological applications of high resolution TEM for 3D structure determination include electron crystallography (Glaeser, 1999), single-particle reconstruction for macromolecules and their assemblies (Frank, 2002), and helical reconstruction for structures with repeating helical subunits (DeRosier et al., 1999). These techniques have provided a wealth of knowledge in the structural biology community, a small sampling of which includes the atomic structure of the tubulin dimer (Nogales et al., 1998), the structure of bacterial flagellar motors (Francis et al., 1994), the architecture of the nuclear pore complex (Yang et al., 1998), and a reconstruction of GroEL (Ludtke et al., 2004). To yield a more complete biological view, such high resolution structures, which are often determined using isolated complexes, may be fit and oriented into lower resolution TEM images by a process known as docking (Baker and Johnson, 1996). However, these methods for 3D reconstruction often require the averaging of thousands of similar copies of the structure of interest to generate sufficient resolution. As such, they are unsuitable for studying many subcellular components, such as organelles and membranes, whose structures vary widely even within the same cell (McEwen and Marko, 2001). The desire to study organelles at close to the resolution afforded by these averaging-based approaches combined with the increased availability of high voltage TEMs led to the birth of electron tomography.

The introduction of TEMs that operate at substantially increased voltages allowed microscopists to image thicker sections while still attaining useful contrast and resolution. High accelerating voltages facilitate the imaging of thicker sections by increasing the mean-free path of electrons traveling through the sample and limiting chromatic aberration (Frank, 2008). While some groups have employed ultrahigh voltage electron microscopes (UHVEMs) operating in the megavolt range (Takaoka et al., 2000), most practical

applications involve the use of intermediate voltage electron microscopes (IVEMs) operating in the 300-400 kV range. Unfortunately, even when using IVEMs, electron micrographs of samples with a thickness of greater than a few hundred nanometers tend to be difficult to analyze due to the confounding effect that features from all depths of the sample are superimposed onto the same 2D projection. This results in micrographs that appear blurred or smeared, especially in the vicinity of objects whose topologies change significantly throughout the depth of the sample.

To circumvent this issue and produce useful reconstructions of subcellular components using IVEMs without the need for ssTEM, researchers adapted the principles of x-ray computed tomography (CT) to the electron microscope (DeRosier and Klug, 1968). The resulting technique, ET, enables the reconstruction of a 3D digital volume with isotropic pixel dimensions from a set of 2D projections of the sample acquired at different orientations with respect to the primary electron beam. In a typical ET experiment, this set of projections, called a tilt series, is acquired by tilting and imaging the sample in angular increments about an axis perpendicular to the electron beam. Reconstruction is possible because, assuming that the electron paths are known or can be estimated, the only unknown is the manner in which the sample's density is distributed across the imaging plane (McEwen and Marko, 2001). This unknown distribution can be computed by an algorithm known as back-projection, in which the known densities of a given projection image are distributed evenly over rays that re-trace the imaging path. This is repeated for each image in the tilt series, and as the rays from all projections intersect, they sum to form a 3D reconstruction of the original sample.

In practice, tilt series are recorded by rotating the sample over a range of  $\pm 60$ - $70^\circ$  in  $1$ - $2^\circ$  increments using a computer-controlled goniometer. Though progress is being made (Palmer and Löwe, 2014), rotations outside of this range are typically not attainable

due to obstruction of the electron beam by the specimen holder at angles approaching 90°. This results in a so-called “missing wedge” of information corresponding to the angular range between the maximum tilt angle and 90° (Frey et al., 2005). The missing wedge has the deleterious effect of reducing the axial resolution of the final reconstruction, and may introduce biases to post-reconstruction analyses such as sub-tomogram averaging (Nickell et al., 2005; Nicastro et al., 2006). Fortunately, the size of the missing wedge can be reduced by collecting an initial tilt series, rotating the sample by 90° about the TEM’s optical axis, collecting a second tilt series, and computationally combining both reconstructions. This process, known as dual-axis ET (Mastronarde, 1997), improves the resolution of the output tomogram at the expense of increased specimen damage as well as increased collection and processing times. Recent works have demonstrated that the missing wedge can be further reduced by combining reconstructions from more than two axes in an analogous manner (Ellisman et al., 2014). Additionally, methods such as high pressure freezing and freeze substitution can enhance ultrastructural preservation for ET and result in higher quality tomograms (Sosinsky et al., 2008).

Since each successive tilt angle changes the working thickness of the sample that the electron beam must penetrate, the focus and positioning of the TEM have to be slightly adjusted in between collecting each projection image. Traditionally, this meant that tilt series collection was extremely labor intensive; a human operator had to be present at the TEM to rotate the goniometer, manually adjust alignments, collect an image, and repeat. However, computational advances have allowed for software that can interface with the TEM and automatically adjust these parameters in between each tilt (Mastronarde, 2003). After imaging, the tilt series must be post-processed in a number of ways before a successful reconstruction can be attained. The most important of these steps is the computational alignment of successive projection images with one another, and this



alignment is typically performed with the assistance of fiducial markers in the form of gold nanoparticles (AuNPs) that are fixed to the top and bottom of the sample prior to imaging. A number of software packages have been developed for tilt series post-processing and reconstruction, and include IMOD (Kremer et al., 1996), TOM (Nickell et al., 2005), EM3D (Harlow et al., 2001), TomoJ (Messaoudil et al., 2012), and TxBR (Lawrence et al., 2006; Phan et al., 2012).

Perhaps not surprisingly, the field of ET progressed towards the reconstruction and stacking of tomograms from serial thick sections, a technique known as serial section electron tomography (ssET) (Soto et al., 1994). In ssET, tilt series are acquired for each section and individually reconstructed. This series of reconstructions is then stacked together by computationally aligning user-placed fiducial marks between the last and first tomographic slices from consecutive reconstructions. Though extremely laborious, ssTEM has been successfully applied to build high resolution reconstructions of the Golgi apparatus (Ladinsky et al., 1999), the node of Ranvier (Sosinsky et al., 2005) and even full cell models of pancreatic beta cells (Noske et al., 2008) and hair stereocilia (Vranceanu et al., 2012).

Despite the aforementioned successes, the reconstruction of even partial neurons by ssTEM is currently infeasible if any structures in addition to the soma are desired. From personal experience, the automated collection of a single tilt series using a JEOL JEM 4000EX with two degree increments takes roughly one hour, without accounting for the time required for TEM startup and alignment. Fiducial tracking for tilt series alignment may take an additional two to three hours per tilt series, a figure that is highly dependent upon the image quality of the tilt series and the experience of the researcher. The two whole-cell reconstructions of Noske and colleagues necessitated 46 and 27 sections cut to thicknesses of 300-400 nm (Noske et al., 2008). Such reconstructions were possible due

in large part to the roughly spherical nature of pancreatic beta cells; tracking the same cell from section to section was trivial. Now, consider a neuron, whose neurites may branch from 10s to 100s of microns away from its soma after only a few steps through the stack of serial sections. Such branched processes would be impossible to find on the TEM when looking in axial increments of 300-400 nm; the microscopist simply wouldn't know which region of the sample to image.

In spite of this, ET maintains an important presence in the toolbox of modalities for large-scale structural studies of the brain (Vihinen et al., 2013). As previously discussed, ET is employed in this dissertation to provide a high resolution complement to SBEM. Whereas SBEM is preferable for the determination of gross morphological parameters and distributions of organelles across whole neurons, ET can be used to correlatively address questions at the single organelle level, such as the distribution of nuclear pores or the geometry of mitochondrial cristae.

### **1.7. Array Tomography**

A final modality in the toolbox for large-scale neuronal reconstructions is array tomography (AT), an all-encompassing term used to describe a variety of methods based on the collection of a 3D volume from serially sectioned tissue using an SEM. Unlike the serial-block face techniques, sections for AT are cut and collected on a substrate prior to insertion into the chamber of the SEM for imaging (Wacker and Schroeder, 2013). AT, therefore, has more in common with ssTEM and the serial-block face imaging techniques than it does ET; the word "tomography" is used here to refer to physical rather than computational slicing.

AT was first introduced to the biological sciences by Micheva and Smith as an approach for CLEM (Micheva and Smith, 2007). Driven by a desire to achieve axial

resolutions better than those afforded by confocal microscopy, Micheva and Smith turned to ultrathin cryosections, which provide better resolution than thicker, conventional cryostat sections (Mori et al., 2006). Such serial ultrathin sections were sequentially imaged to yield 3D distributions of fluorescent antibodies, and could then be cyclically eluted, re-labeled, and re-imaged with a different set of antibodies. Furthermore, by the use of LRWhite embedding media (Newman and Hobot, 1999), these same sections were capable of being post-stained with heavy metals and stably imaged in the SEM using the detection of BSE signals. In this way, Micheva and Smith were able to produce overlays of fluorescent signals on EM images with reasonably maintained ultrastructure. Such methods, however, are of course a trade-off between fluorescence preservation and the maintenance of tissue ultrastructure at the EM level. Oberti and colleagues were able to achieve better ultrastructural preservation and regain some fluorescence lost during the embedding process by the application of anti-dye antibodies (Oberti et al., 2011). A subsequent study introduced super-resolution imaging to AT, implementing direct stochastic optical reconstruction microscopy (dSTORM) to yield 28 nm lateral resolution (Nanguneri, et al., 2012).

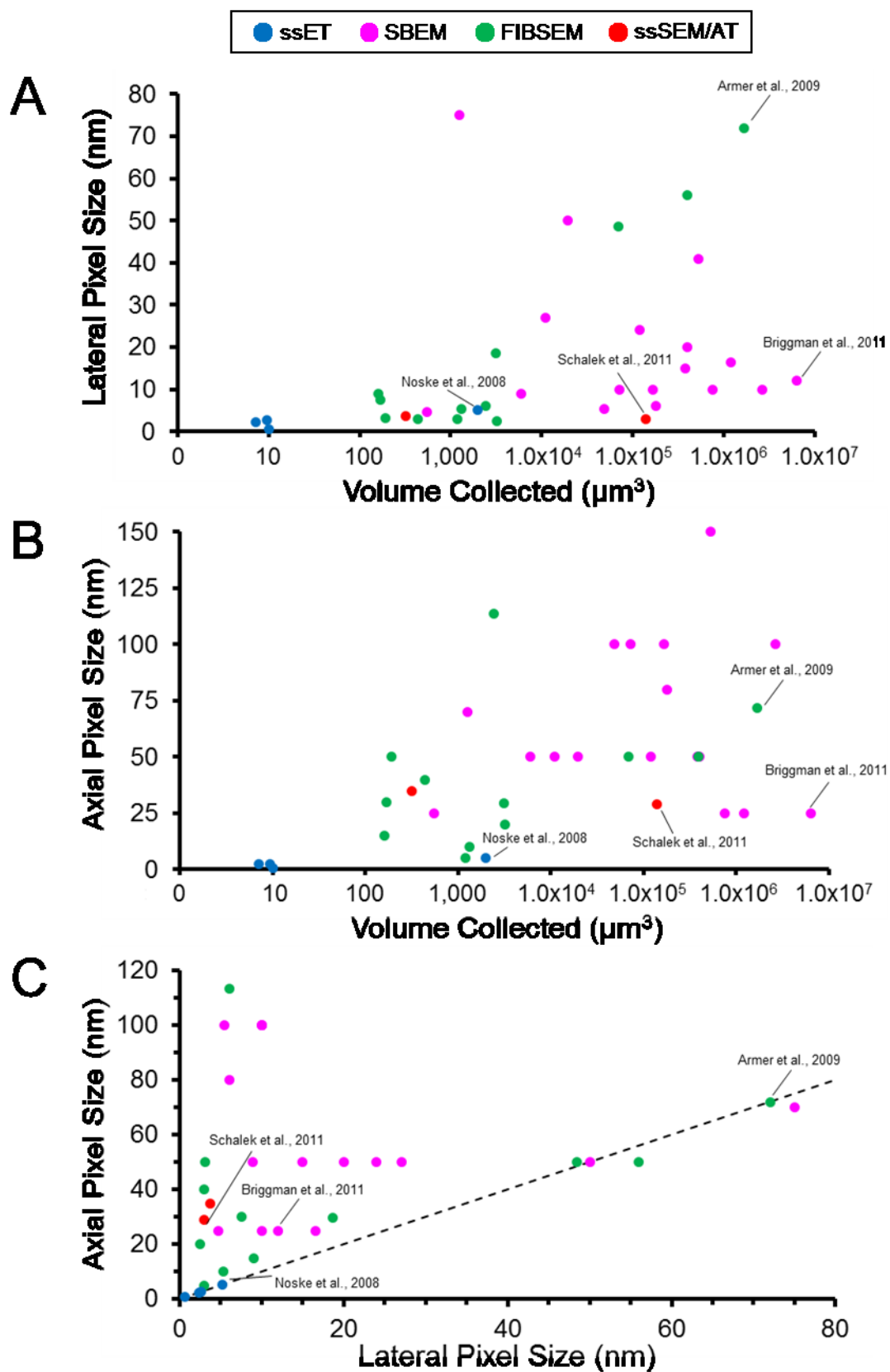
An alternative approach to AT is to forego antibody labeling in favor of acquiring large SEM volumes of thinner sections with better ultrastructural preservation (Horstmann et al., 2012). Such an approach represents a potential improvement over the aforementioned large-scale EM modalities in a number of ways. First, the use of an SEM with BSE detection eliminates the need to place sections on the less stable, electron transmissive substrates that would be required for ssTEM. Secondly, unlike the serial-block-face modalities, AT is non-destructive; sections are not significantly damaged after imaging and can be re-imaged at different magnifications or configurations if desired. To compete with the high throughputs of the serial block-face modalities, however, the

process of section cutting and collecting in such a scheme must be automated. A device capable of this automation, the automatic tape-collecting ultramicrotome (ATUM), was introduced by Kenneth Hayworth and Jeff Lichtman in 2006. The ATUM is designed to cut ultrathin sections of 30-35 nm and automatically collect them from the knife's waterboat onto specially designed copper tape fed by a conveyor belt (Hayworth et al., 2006; Hayworth et al., 2014). A recent report demonstrated the power of this approach, citing the collection of 2,100 consecutive, 29 nm-thick sections using the ATUM (Schalek et al., 2011). After sectioning, the section-containing copper tapes are then cut and placed onto silicon wafers that contain on the order of 100s of ultrathin sections, generating so-called ultrathin section libraries (UTSLs; Hayworth et al., 2014). Individual sections from each UTSL can then be sequentially imaged by BSE detection in an SEM. This approach confers the additional advantage of reducing the time required for image collection, since individual UTSLs can in theory be imaged in parallel on multiple SEMs. Parallelization of this degree is not currently possible using the serial block-face modalities, which must physically remove a section before the next one is revealed. Despite the power of this technique, its applications in the field have so far been limited, likely due in large part to its reliance on highly specialized, custom-built machines (Peddie and Collinson, 2014).

## **1.8. Discussion and future perspectives**

Advances in technologies for specimen preparation, imaging, data storage, and data analysis have fueled a renaissance in the field of quantitative 3D EM. Data obtained from modalities such as SBEM provide unprecedented volumetric snapshots of the *in situ* biological organization of the mammalian brain across a multitude of scales (Figure 1.2). When optimized staining protocols are employed (Deerinck et al., 2010), the resultant datasets possess enough breadth of field and resolution to be mined for answers to a

**Figure 1.2. A comparison of the pixel sizes and total image volumes used in a sampling of publications that employed 3D EM methods.** The reported lateral pixel sizes, axial pixel sizes, and total volumes are compared for studies published using four 3D EM modalities: ssET, SBEM, FIBSEM, and AT. A study of interest from each modality is denoted by its citation on all graphs. In (A) and (B), lateral and axial pixel sizes are compared to total volume, where total volume is plotted on a log scale. With the notable exception of the study by Noske and colleagues, which achieved a total volume comparable to that of many FIBSEM reports, ssET studies demonstrate the finest pixel sizes but smallest total volumes. On the other end of the spectrum, SBEM has been the modality of choice for most high-volume studies, with the notable exception of the large-scale volumetric FIBSEM dataset ( $\sim 1.7 \times 10^6 \mu\text{m}^3$ ) collected by Armer and coworkers. Most high-volume studies have, however, come at the expense of resolution in either the lateral or axial directions, or in some cases, both. The SBEM work reported by Briggman and colleagues represents one example of a massive dataset that was still collected at relatively fine pixel sizes (12 nm lateral, 25 nm axial,  $\sim 6.3 \times 10^6 \mu\text{m}^3$  volume). A comparison of reported lateral and axial pixel sizes (C) demonstrates that very few large-scale 3D EM studies have utilized isotropic voxel dimensions (isotropy is indicated by the dashed line). Outside of ssET, FIBSEM studies are most likely to utilize near-isotropic voxels, while a few SBEM studies have reported isotropic voxels by utilizing coarse lateral pixel sizes to closely match their axial step sizes.



number of biologically relevant hypotheses. A single dataset, for example, could be used to simultaneously map the distribution of synapses (Kreshuk et al., 2011; Morales et al., 2011; Kreshuk et al., 2014; Plaza et al., 2014; Staffler et al., 2014), quantify the morphologies of dendritic spines (Wilke et al., 2013; Wilke et al., 2014), explore the ultrastructure and distribution of organelle networks (Kalsen et al., 2011; Motskin et al., 2011; Hatori et al., 2012; Zhuravleva et al., 2012), analyze the composition of chromatin (Rouquette et al., 2009), and establish connectivities between neurons (Briggman et al., 2011; Helmstaedter et al., 2013; Kim et al., 2014). Historically, however, most 3D EM datasets have been collected with only a very specific scientific goal in mind. Such datasets are used to extract the desired images or quantities and are then sent to archival storage, where they are oftentimes forgotten and never looked at again. As a consequence, there already exists a wealth of archived data that could be re-analyzed to answer a host of other intriguing scientific questions. This amount of data is only growing; microscopes are already capable of collecting terabytes of image data per day, and this number will soon grow as technologies such as an SEM capable of imaging with 61 beams in parallel become adopted (Marx et al., 2013; Keller et al., 2014). Therefore, it is abundantly clear that in order for such analyses to be feasible and desirable to the average scientist, rapid and largely automated tools for analysis are needed.

Although these relatively new modalities are flexible enough to enable studies with diverse biological goals, advances in the fields of segmentation and data analysis remain largely driven by the pursuit of connectomics (Kleinfeld et al., 2011; Lichtman and Denk, 2011; Briggman and Bock, 2012; Plaza et al., 2014). With an eye towards this goal, semi-automatic segmentation algorithms capable of labeling and tracking individual fibers through dense tangles of neuropil are currently a major focus of the community (Jurrus et al., 2009; Straehle et al., 2011; Andres et al., 2012; Liu et al., 2013). Though analogous

algorithms have been developed for organelles such as mitochondria (Giuly et al., 2012; Lucchi et al., 2012; Seyedhosseini et al., 2013a) and nuclei (Jaume et al., 2012), published reports of their applications to full, large-scale datasets are few and far between (Tek et al., 2014). Despite their obvious biological importance, organelles and other constituents of the subcellular compartment have been largely ignored at the quantitative level in state-of-the-art, large-scale 3D EM reconstructions. Indeed, some connectome-centric SBEM datasets have even utilized staining protocols specifically designed to leave the subcellular compartment unlabeled (Briggman et al., 2011; Helmstaedter et al., 2013).

In light of this, it is evident that there remains a substantial, and predominantly untapped, opportunity for the application of serial block-face imaging modalities to the study of subcellular structure-function relationships across large scales. Data resulting from such efforts could be viewed as complementary to the endeavors of the connectomics community, and the combination of reconstructions from both would yield a more complete picture of cellular neuroanatomy. With this in mind, one of the major goals of the work presented in this dissertation was to establish a workflow for the semi-automatic segmentation and morphological characterization of organelles in large-scale 3D EM datasets. In Chapter 2, the work leading to the development of such a workflow will be described. An architecture for achieving accurate segmentations of organelles will be outlined, and the ability to scale this architecture to large SBEM datasets will be demonstrated. Chapter 3 will introduce streamlined methods for enhancing and quantifying the morphologies and 3D distributions of automatically segmented structures. Finally, Chapter 4 will implement these previously developed technologies to study a biologically intriguing question, namely, the chromorphology of organelles in the suprachiasmatic nucleus.



## **Chapter 2**

# **The Automatic Segmentation of Multi-scale Neuroanatomical Features in 3D EM Image Stacks**

## **2.1. Introduction**

The generation of models from a series of 2D observations has long been recognized as the rate-limiting step of quantitative 3D EM. Accounting for a span of time ranging from the first microscopic observations of van Leeuwenhoek to the modern age of EM automation, Stephen Senft recently wrote that, for the neurosciences, the “bottleneck to reconstruction from each epoch has been processing speed, data access..., and most importantly, processing intelligence” (Senft, 2011). Fortunately, the specimen preparation and instrumentation advances discussed in Chapter 1 have combined to address one of these roadblocks, furnishing neuroscientists with unparalleled access to vast amounts of microscopic data. The last two impediments, processing speed and intelligence, remain topics in need of advancement, and both will be addressed in part by the technologies developed and described throughout the remainder of this dissertation.

Before the introduction of these technologies, however, this chapter will begin with a brief historical discussion of the methods that have been employed by the community to furnish accurate 3D models. Following this will be a review of contributions towards automating these processes, and a discussion of the current state of the field.

### **2.1.1. The manual segmentation bottleneck**

Prior to the advent of modern computers, researchers resorted to a host of innovative, yet extremely labor-intensive methods for the construction of physical models from ssTEM micrographs. Early experimenters manually traced structures onto sheets of transparent cellophane, which were then serially aligned and glued together to provide 3D information (Bang and Bang, 1957; Sjöstrand, 1958). Subsequent researchers used photographic enlargers to project the negative of each micrograph onto white cardboard. Structures of interest were then hand-drawn onto the cardboard, cut out, and glued

together to form 3D models (Sotelo et al., 1973). Conceptually similar methods involved stacking cut-outs of polystyrene (Pedlar and Tilly, 1966) and graph paper (Hoffman and Avers, 1973), as well as segments of string wrapped around the perimeter of the structure of interest (Braverman and Keh-Yen, 1983).

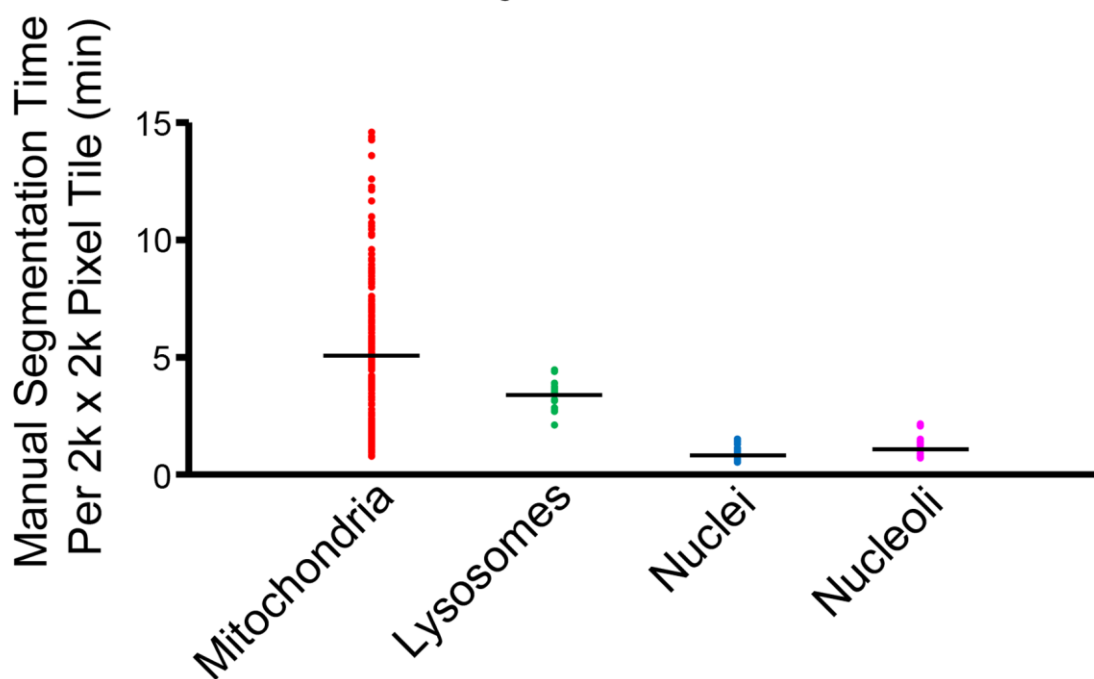
Though such models facilitated easier views of structure than the set of 2D images from which they were derived, they remained bounded by the restraints of the physical world; a model could only grow so large before it would require supports to prevent it from toppling over like a tower of Jenga blocks. Furthermore, such models required physical storage space and were difficult to morphologically quantify with a high degree of precision. As technology advanced and more ambitious imaging projects were initiated, the need for computer-based reconstruction systems became clear (Levinthal and Ware, 1972). Since early microcomputers did not readily permit the simultaneous display of EM images and collection of user input, preliminary systems consisted of digital microcomputers that received user input from drawing tablets interfaced with optical devices (Cowan and Wann, 1973; Wiley et al., 1973; Fox et al., 1975; Moens and Moens, 1981; Prothero and Prothero, 1982). Using such systems, 3D renderings and basic morphological parameters could be calculated and output to a connected oscilloscope or plotter (Macagno et al., 1979). To accelerate the analysis of large stacks of micrographs, systems were built in which serial images could be manually aligned to one another and analyzed as they were automatically played as a filmstrip on a TV camera (Harris and Stevens, 1988). During the mapping of the *C. elegans* connectome, a process that spanned many years, some attempts at utilizing computer-aided reconstruction were made (Stevens and White, 1979; White et al., 1986). Computerized systems, however, remained in their formative stages at this time. Therefore, much of the work was still performed by manually annotating neuronal processes on prints with drafting pens.

Over the next decade, improvements in computer hardware brought with them a number of upgraded systems and software packages for the manual reconstruction and visualization of objects from EM micrographs (Young et al., 1987; Allen and Levinthal, 1990). The SYNU software package facilitated the classification and computation of 3D meshes from segmentations (Hessler et al., 1992). It also permitted the simultaneous display of numerous meshed objects and enabled the user to create high resolution movies from distinct orientations. The IMOD software package, which was introduced in 1996 and still receives frequent use today, allows microscopists to segment structures of interest by drawing contours around them using a mouse or other input device (Kremer et al., 1996). Such contours are sorted into hierarchical objects that can be individually meshed, displayed, and morphologically quantified. IMOD utilizes the MRC image format (Crowther et al., 1996), which provides the advantage of appending all micrographs from a series into one file as a 3D stack. Such single files are much easier to store, keep track of, and generate models from than a series of thousands of individual files. Since the introduction of IMOD, a multitude of other software packages for the manual segmentation and reconstruction of objects in 3D EM datasets have been introduced, including Xvotrace (Perkins et al., 1997), Bsoft (Heymann, 2001), UCSF Chimera (Pettersen et al., 2004), XMIPP (Sorzano et al., 2004), Reconstruct (Fiala, 2005), AUTO3DEM (Yan et al., 2007), EMAN2 (Tang et al., 2007), Viking (Anderson et al., 2010), KNOSSOS (Briggman et al., 2011), and TrakEM2 (Cardona et al., 2012). This list will surely continue to grow as the needs of the community expand and evolve.

Despite the plethora of software options currently available to the field, human-based manual annotations remain necessary to ensure accurate reconstructions. As an example, consider the reconstruction of a neuron and all of its processes across 1,000 SBEM slices. Even if an automated algorithm could achieve an accuracy of 99.9% for the

segmentation of this neuron (a value that is substantially better than any such algorithm has ever performed), it would still almost certainly produce at least one fatal error in the form of a falsely merged or separated neurite. Researchers, therefore, are typically presented with three options when confronted with the need for a reconstructed model: (1) use an automatic algorithm and accept its associated errors, (2) apply an automatic algorithm and manually correct the errors of its output, or (3) perform purely manual segmentation. Unfortunately, the first two approaches tend to be ignored by all but a few groups, since most automatic approaches are not readily accessible or easily implementable on the systems of a standard lab.

Most neuroscientists, therefore, routinely opt for the manual segmentation option. Though such a choice may be serviceable for small studies, it essentially precludes the completion of any large-scale reconstructions in a timely manner. Even when using expedited methods and shortcuts such as skipping slices or approximating organelles as spheres or cylinders, manual segmentation remains a laborious endeavor (Noske et al., 2008). Reconstructing neurons from even small regions of the *Drosophila melanogaster* visual system has been reported to take several months to years in terms of labor (Chklovskii et al., 2010; Plaza et al., 2012). The manual segmentation of mitochondria on all slices of an SBEM dataset the size of the one used in this chapter ( $\sim 450,000 \mu\text{m}^3$ ) would require an estimated 2.3 years of work (Figure 2.1). This means that manually reconstructing all mitochondria from a dataset the size of a full mouse brain ( $\sim 500 \text{ mm}^3$ ) would necessitate about one year of nonstop work from every citizen in the city of Chicago ( $\sim 2.7$  million people). An analogous effort for the human brain would require the same workload from every person living on the continents of Asia and Africa combined ( $\sim 5.5$  billion people).



**Figure 2.1. The manual segmentation of organelles from SBEM image stacks represents a significant bottleneck to quantitative analyses.** A scatter plot of the amount of time required for a highly trained neuroanatomist to segment all instances of a specific organelle in SBEM tiles of size 2,000 x 2,000 pixels demonstrates this impediment. Average values are represented by horizontal bars (mitochondria = 5.01 min., lysosomes = 3.43 min., nuclei = 0.93 min., nucleoli = 1.24 min.). Since mitochondria are ubiquitously present throughout most tissues, extrapolation of their average segmentation time per tile to the size of a full dataset can reliably predict the actual segmentation time required for such a volume. For a dataset the size of the one used in this report (stack volume ~ 450,000  $\mu\text{m}^3$ , tile size ~ 60  $\mu\text{m}^2$ ), the manual segmentation of all mitochondria would require roughly 2.3 years, placing it well outside the realm of feasibility. This effect is further exacerbated when experiments requiring segmentations from SBEM stacks over multiple samples or experimental conditions are desired.

As outlandish as these hypothetical efforts may seem, this sort of outsourcing has already been attempted, albeit on much smaller scales. In their partial reconstruction of the retinal wiring diagram, Kevin Briggman and Winfried Denk employed over 200 segmenters to trace skeletons through neuronal processes (Briggman et al., 2011). As a first attempt at achieving large-scale reconstructions of organelles, I initially employed a similar method with a group of undergraduate student volunteers. In collaboration with Monica Berlanga, volunteers were recruited via job postings to the UCSD Career Services website and trained on-site to segment using IMOD. To overcome resource limitations and create a flexible work environment, the trained volunteers installed the IMOD software on their own personal computers and segmented remotely. For remote segmentation to be possible, however, SBEM stacks first had to be decomposed into smaller sub-stacks that could reasonably fit into the hard disk and memory of a standard laptop computer. Thus, each image stack was first manually inspected for features of interest, which were then extracted to sub-volumes precisely large enough to contain the feature of interest in its entirety. The size of a typical sub-volume was ~1-2 GB, which is small enough for even outdated laptop computers to reasonably handle. A diagram of this workflow is shown in Figure 2.2.

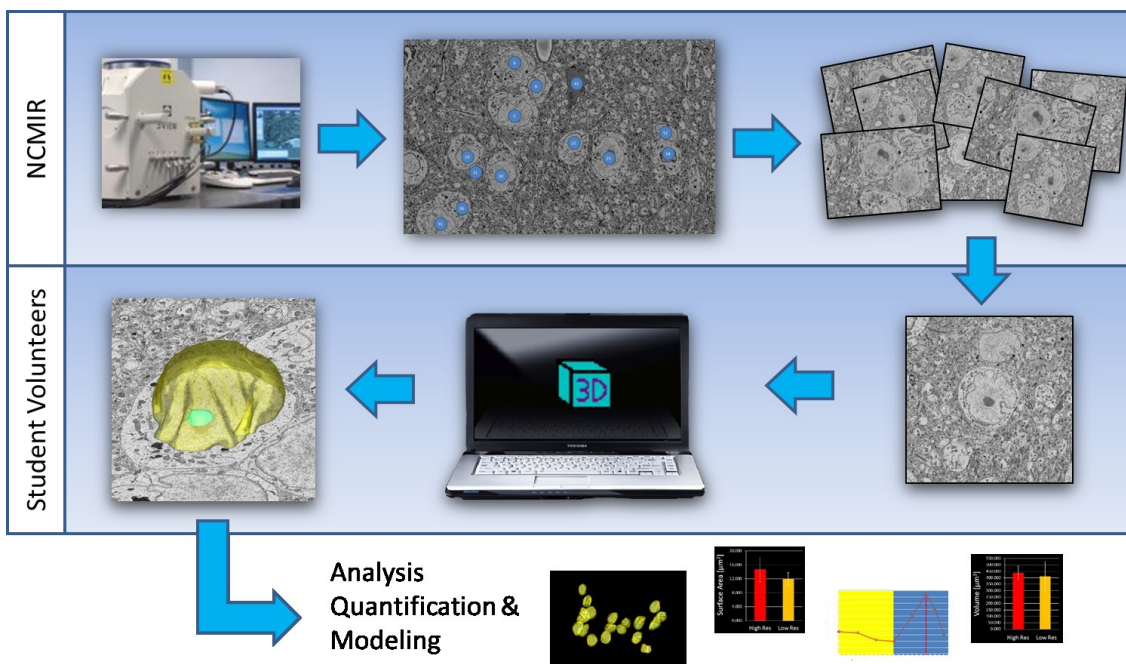
To organize the group, I created a web portal through which volunteers could download sub-volumes to work on, upload model files when their segmentations were complete, and ask questions or receive advice. The portal was created as part of the SLASH (Scalable system for large data analysis and segmentation utilizing a hybrid approach) group's Google website, and files were distributed using Google Drive. The portal was password protected so that files were only available for download by team members. When logged in, seven tabs were available for volunteers to access (Figure 2.3), the purposes of which are described here:

1. About – Contained links that introduced the volunteers to concepts relevant to the project, such as SBEM and the SCN.
2. Calendar – Provided dates for deadlines, training days, and open office hours.
3. Challenges – Provided details specific to the current segmentation task, including the nature of the feature of interest, the pixel dimensions of the current sub-volumes, and the desired colors, numbers, and names for IMOD objects (Figure 2.4).
4. Discussion Board – Provided an interactive environment in which volunteers could ask and answer questions pertaining to the current segmentation task (Figure 2.5).
5. The Team – Contained the names and contact information of all volunteers.
6. Tutorials – Contained links to tutorials and instructional videos for segmentation using IMOD.
7. Upload – Allowed the volunteers to upload their completed model files to a shared Google Drive folder.

This portal was vital for facilitating the remote segmentation process, since it afforded volunteers an easy way to exchange files without the need to physically come into the lab.

The results obtained from this so-called “collaborative segmentation” endeavor were critical in establishing the need for a more powerful analysis method. Though such collaborative efforts have proven to yield satisfactory segmentations, they are clearly not viable, long-term solutions to the manual segmentation bottleneck. First, the quality of work from volunteers cannot always be trusted without substantial post-verification or redundancy. Furthermore, the task of manual segmentation is an acquired taste; many volunteers do not enjoy the work and quickly lose the desire to contribute. Finally, volunteers need to be recruited, trained, and coordinated, and each of these tasks require substantial investments on the part of the primary investigator.





**Figure 2.2. The collaborative segmentation workflow.** After imaging, alignment, and downsampling, each dataset was manually inspected for instances of the feature of interest, in this case nuclei, that were fully contained within the volume. Each instance was assigned a numerical value and extracted to individual subvolumes using the IMOD programs *trimvol* and *boxstartend*. These subvolumes were then distributed to trained volunteers for segmentation and analysis on their own workstations. Subvolumes were generally limited to sizes of no larger than 1 GB to keep them manageable for personal computers and laptops. After completion, volunteers uploaded their segmentations to a shared Google Drive folder via a custom-designed web portal. Completed segmentations were subsequently downloaded, checked for accuracy, and added to the pipeline for morphological analysis and quantification.

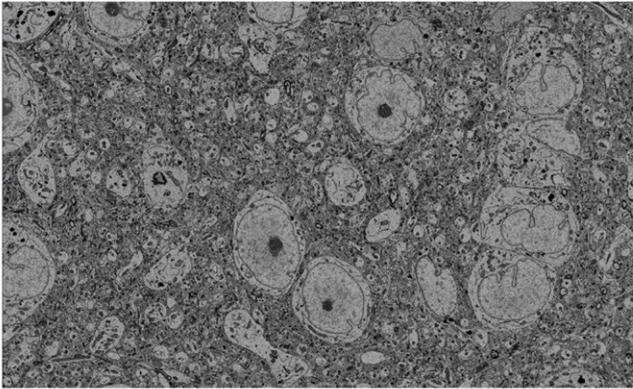
Home

- \*METHODOLOGY\*
- About
- Affiliated Groups
  - Team MICRO
    - About
    - Calendar
    - Challenges
    - Discussion Board
    - Discussion Board (WIB)
    - The Team
    - Tutorials
    - Upload
- Announcements
  - [Untitled]
  - [Untitled]
  - SLASH IMOD Stereology Plugin Added
  - WIB tool updated, 3D Electron Microscopy Tracing Demo Video
- Articles
- Download Tools
  - Automatic Cell Segmentation
  - Cell Tracer with Mechanical Turk
  - Cyloseg
  - IMOD Plugins
    - Bead Helper Plugin
    - Drawing Tools Plugin
    - Expert Segmentation
    - Finding Round Compartments
    - Interpolator Plugin
    - Livewire
    - Name Wizard Plugin
    - Stereology Plugin
    - Stereology Plugin Tutorial
    - Online Segmentation Tools
    - SLASH 3D Model Viewer
  - WIB
    - WIB: Feature Outline
- Forum and Contact
  - The Team
- Segmentation Algorithms
  - Series-ANN

**SLASH**

**Team MICRO**

## Microscopic Investigations of Circadian Rhythms & Oscillations



Team MICRO is an interdisciplinary group of researchers affiliated with the [National Center for Microscopy & Imaging Research \(NCMIR\)](#) at UCSD. Our goal is the accurate annotation of terabyte-sized serial block-face SEM datasets for the quantification of daily ultrastructural oscillations at the organellar level. To achieve this goal, we employ a method known as **collaborative segmentation**, which involves the dissemination of bite-sized pieces of the puzzle to expertly trained team members for remote segmentation. In addition to aiding in the exploration of fundamental physiological processes, our output is also being used to drive and train the development of autosegmentation algorithms and machine-learning approaches to image annotation.

**For Group Members:**  
Please [click here](#) to log in using your Google account to gain access our private pages.

**Figure 2.3. An online portal to facilitate collaborative segmentation.** The password-protected portal was designed as a sub-group under the main SLASH website. Each volunteer was granted access to the portal by signing in via their Google account. Once inside the portal, volunteers could access a calendar of upcoming deadlines, read instructions for the current task, ask or answer questions on the discussion board, and upload completed segmentations to a shared Google Drive folder.

← → ↻ <https://sites.google.com/site/slashsegmentation/groups/team-micro/challenges> ☆ 🔍 🗨️ 📄 ☰

according to the Circadian cycle.

**Project Details:** Each item will be given three subvolumes containing one full nucleus each. Each subvolume should be traced and should have its own model file, with each model file containing at least two objects as described below:

**Steps for analyzing each subvolume:**

1. Open your subvolume in IMOD.
2. From the Special menu, open both 'Drawing Tools' and 'Interpolator'. You will leave these open in the background at all times while tracing. You should not need to adjust any of the options in the interpolator window.
3. Identify your nucleus. Each volume has only one FULLY CONTAINED nucleus. This means that when you pan through the volume using the Page-Up and Page-Down keys, only one nucleus will be there from start to end. If you press the Insert key, the nucleus that should be segmented will be in the middle of the image. Pan through the volume using the Page-Up and Page-Down keys to ensure that it is fully contained. This is the nucleus you will be tracing.
4. Determine how many nucleoli are present in your nucleus. The nuclei we are tracing contain anywhere from 1-4 nucleoli. Pan through your whole nucleus and count how many nucleoli are present. We will show you how to identify nucleoli.
5. Create the appropriate number of objects. Remember that you will need one object for each organelle being traced. Thus, if your nucleus has two nucleoli, you will need 3 objects: one for the nucleus, and one for each nucleolus.
  - a. Create a new object by selecting: Edit → Object → New
  - b. Each time you do this, you will see the object count in the 3dmod window increments by one.
6. The naming and coloring conventions for objects are as follows:
 

Object #	Object Name	Red	Green	Blue	Contour Type	Color
1	nuclear membrane	255	255	0	Closed	Yellow
2	nucleolus	0	255	255	Closed	Cyan
7. In the case where your cell contains more than one nucleolus, you should create a separate object for each one, and number the objects consecutively. For example, if your cell has two nucleoli, your objects would be:
 

Object #	Object Name	Red	Green	Blue	Contour Type	Color
1	nuclear_membrane	255	255	0	Closed	Yellow
2	nucleolus1	0	255	255	Closed	Cyan
3	nucleolus2	0	255	255	Closed	Cyan
8. Set the correct pixel size and Z-scale values. If you notice your model looks weird in model view mode, it is because these values were not set correctly. Click Edit → Model → Header
  - a. Added Z-scale = 1
  - b. Total Z-Scale = 1.9233
  - c. Resolution = 3
  - d. Pixel Size = 15.5979 nm/Notes:

**Notes:**

- (1) The resolution of these datasets is not high enough to discern between the inner and outer nuclear membranes. Therefore, we will trace only one object called the 'Nuclear Membrane along the perimeter of the nucleus.
- (2) As mentioned in the background, regions of ER are continuous with the nuclear membrane. These should NOT be included in your trace.
- (3) The nucleus to be traced will be the ONLY nucleus that is fully contained in

**Figure 2.4.** The instructions given to volunteers for the task of segmenting nuclei and nucleoli. In addition to being trained in person, volunteers were able to access instructions for the current segmentation task from the 'Challenges' tab of the main portal. The instructions included detailed information on how to identify the structure of interest, how to generate the proper IMOD model file, and the naming and color conventions for each object. Shown here is an excerpt of the instructions given for the segmentation of nuclei and nucleoli, as visualized from within the portal.

SLASH

Home  
 \*METHODOLOGY\*  
 About  
 Affiliated Groups  
 Team MICRO  
 About  
 Calendar  
 Challenges  
 Discussion Board  
 Discussion Board (WIB)  
 The Team  
 Tutorials  
 Upload  
 Announcements  
 [Untitled]  
 [Untitled]  
 SLASH IMOD Stereology Plugin Added  
 WIB tool updated, 3D Electron Microscopy Tracing Demo Video  
 Articles  
 Download Tools  
 Automatic Cell Segmentation  
 Cell Tracer with Mechanical Turk  
 Cytoseg  
 IMOD Plugins  
 Bead Helper Plugin  
 Drawing Tools Plugin  
 Expert Segmentation  
 Finding Round Compartments  
 Interpolator Plugin  
 Livewire  
 Name Wizard Plugin  
 Stereology Plugin  
 Introduction  
 Stereology Plugin Tutorial  
 Online Segmentation Tools  
 SLASH 3D Model Viewer  
 WIB  
 WIB: Feature Outline  
 Forum and Contact  
 The Team  
 Segmentation Algorithms  
 See: ANN

Affiliated Groups > Team MICRO >  
 Discussion Board

MICRO  
 Microscopic Investigations of Circadian Rhythms & Oscillations

Discussion Board

NEW TOPIC [Refresh] Mark all as read Actions Filters Help

[Search]

MICRO at NCMIR Shared privately Membership and email settings  
 13 of 13 topics (13 unread) Members About

Welcome to the MICRO Discussion Board! Please use this forum to ask and answer questions about segmenting, identifying structures, and any other issues you may have during your internship at NCMIR.

Edit welcome message Clear welcome message

<input type="checkbox"/>		First Deadline & File Uploading for MICRO (1) By me - 1 post - 6 views	10/23/12
<input type="checkbox"/>		(4) By [redacted] - 4 posts - 3 views	5/12/12
<input type="checkbox"/>		(2) By [redacted] - 2 posts - 3 views	5/1/12
<input type="checkbox"/>		stigmoid body? (11) By [redacted] - 11 posts - 6 views	1/4/12
<input type="checkbox"/>		Nucleus membrane segmenting (7) By [redacted] - 7 posts - 4 views	12/14/11
<input type="checkbox"/>		Line Thickening (7) By [redacted] - 7 posts - 9 views	11/8/11

**Figure 2.5. A discussion board for volunteers to ask and answer questions encountered while segmenting.** When volunteers encountered a problem related to the IMOD segmentation software or their specific subvolume, they were asked to post the question to an interactive discussion board within the portal (names have been blacked out to protect identities). This scheme conferred two advantages: (1) questions could be answered by myself or another volunteer in a timely manner, and (2) such answers would be preserved in case other volunteers subsequently encountered the same problem.

Most of the aforementioned drawbacks of collaborative segmentation can be circumvented if contributions towards reconstructions are made in parallel by many thousands of laypeople. Such approaches, known as crowd-sourcing, have been increasingly applied towards navigating the bottleneck of manual segmentation (Ball, 2014; Valeo, 2014). Rather than being restricted to highly trained experts, crowd-sourcing methodologies leverage upon the recruitment of micro-laborers from the general population. Such micro-laborers are then trained effortlessly with little or no cost to the investigator; training frequently involves the presumptive laborer reading a simple paragraph and producing accurate results on a few test examples, all of which are provided to them automatically by a web interface. Achieving reasonable results with this minimal degree of training is possible because the tasks given to crowd-sourced laborers are relatively simple; instead of being asked to trace a neuron in its entirety, one might simply be asked whether or not two points are located within the same neurite (Giuly et al., 2013). With well-stained material and quality images, such decisions are incredibly easy for a human to make, even with little-to-no practical training.

The most well-known example of crowd-sourcing in the neurosciences is Sebastian Seung's EyeWire (Kim et al., 2014). In this web-based application, users are tasked with tracking the processes of a single neuron through a 3D cube of EM data overlaid with a supervoxel oversegmentation. To keep its users captivated and make them feel as if they are playing a competitive game, EyeWire displays 3D renderings as the player generates them and maintains a ranking system complete with weekly competitions, player levels, and profile badge rewards. According to its website, EyeWire has been used to reconstruct over 100 neurons by 130,000 users based in 145 different countries (<http://eyewire.org>). Brainflight, another attempt at generating crowd-sourced reconstructions by immersing users in a video game-like atmosphere, is currently being

developed by Moritz Helmstaedter's group (<http://brainflight.org>). In this game, which will be released for iOS and Android mobile phones and tablets, users are tasked with correctly linking adjacent supervoxels generated by oversegmentations (Dow et al., 2014). Like EyeWire, Brainflight features enough flashy graphics and reward mechanisms to make its users feel as if they are playing a game rather than tediously correcting segmentations. Though they are certainly helpful, useful scientific contributions can still be aggregated from the crowd without the assistance of game-like environments. One such example is the use of micro-laborers to accurately pick out the spherical objects corresponding to true positives from automatically generated segmentations (Lee, 2013).

Although advances in computational architectures and the introduction of crowd-sourcing techniques have helped to partially expedite the pursuit of large-scale neuroanatomical reconstructions, the manual segmentation bottleneck is still firmly entrenched. Even today, scientists remain far away from developing methodologies that can emulate the collective labor force of billions of humans. Achieving this will require a high degree of automation; even the crowd-sourcing approaches discussed here rely upon accurate automatic segmentations as starting points. Thus, it has become clear that the most important contributions towards achieving this goal will come from improvements in the speed and accuracy of automatic segmentation algorithms. A review of the history and current state of such algorithms will follow.

### **2.1.2. Automatic segmentation algorithms and their applications to the neurosciences**

The advent of computer-based systems for generating manual segmentations brought with it the hope that future researchers might be able to harness the power of these machines to produce such segmentations automatically. In 1979, Macagno,

Levinthal, and Sobel concluded their review of 3D computer reconstruction systems in the neurosciences with the following speculative glance to the future:

...it seems quite clear that the rapid advances in electronics will make more and more automation possible at a reasonable cost. Within a few years we can expect that systems will be available that can easily recognize and digitize some features automatically, leaving for the investigator only those cases that are difficult or ambiguous. (Macagno, Levinthal, and Sobel, 1979)

Although history has proven their estimate of a few years to be overly optimistic, much progress has been made towards achieving this goal. The rest of this section will be dedicated to providing a historical account of this progress as applied to 3D EM datasets, followed by a discussion of the current deficiencies that led to the development of the technologies outlined in this dissertation.

Early reports of automated 3D image segmentation focused primarily on tasks such as separating various tissue types in computed tomography (CT) and magnetic resonance imaging (MRI) scans. A number of techniques were developed to specifically segment images from these modalities, and included innovative boundary detectors (Liu, 1977; Zucker and Hummel, 1981), the marching cubes algorithm (Lorenson and Cline, 1987), and simple intensity thresholding (Höhne et al., 1990). Despite the success of these approaches within the medical community, very few of them proved useful when applied to 3D EM datasets for a number of reasons. First, EM micrographs contain numerous membrane-bound structures whose intensity and texture profiles often overlap extensively. This overlap significantly degrades the quality of results obtainable from any operations based heavily on the histogram of the image. Second, most serial EM modalities are highly anisotropic and may simultaneously yield lateral pixel sizes in the range of 3-50 nm and axial pixel sizes in the range of 30-100 nm. This anisotropy means that features of the same object may appear significantly different from one section to the

next. Finally, even relatively small 3D EM datasets of the brain may contain thousands of fibers with unpredictable orientations. The membrane of a dendrite running perpendicular to the cutting plane often resembles a solid, dark, closed contour. However, after a few axial steps through the depth of the dataset, this same dendrite might bend in such a way that situates it parallel to the cutting plane, giving its membrane the appearance of a more diffuse spread of lighter, unconnected pixels. The design of an algorithm to detect both of these instances as membranes of the same dendrite is no trivial task.

As a result of this complexity, the image processing community was rather delayed in tackling the challenge of developing algorithms for the automatic segmentation of serial EM data. One of the first such attempts was performed by Carlbom and colleagues, who used interactive deformable contours, or snakes (Kass et al., 1988), to segment dendrites from ssTEM images of the rat hippocampus (Carlbom et al., 1994). A similar approach was also used to semi-automatically segment nuclei from EM micrographs of HIV-infected cells (Bron et al., 1994). These works were subsequently improved upon by the introduction of algorithms to better complete neuronal boundaries by minimizing a geodesic function (Vazquez et al., 1998) and improve membrane contrast by optimizing a coherence-enhancing diffusion filter (Tasdizen et al., 2005). This approach is still in use today, and a number of groups have recently reported on the successful use of variations of the deformable, or active, contour approach to segment neural processes (Macke et al., 2008; Jeong et al., 2009; Jurrus et al., 2009). However, since it relies on either user-specified initial contours approximating the object boundary or poorly defined image gradients to drive the segmentation, the deformable contour approach is not ideal in terms of either speed or accuracy.

The deluge of new data introduced by the adoption of automated serial EM techniques such as SBEM brought with it an increased interest in the development of fully



automated segmentation algorithms (Briggman and Denk, 2006). To achieve the goal of reconstructing even a small volume of the brain without employing a city's worth of human laborers, semi-automatic methods such as user-seeded contour evolution are not sufficient. As such, the field of EM image segmentation began to shift towards experimenting with approaches developed by the machine learning and artificial intelligence communities. Upon reviewing the field in 2004, Noël Bonnet predicted that:

...image processing tools will be based on methods originating from the fields of pattern recognition and artificial intelligence. Neural networks and expert systems will play an increasing part. Automatic classification, in the supervised or the unsupervised mode, will become more important when certain experimental techniques currently under development come into routine use. (Bonnet, 2004)

Though machine intelligence had previously been employed to address various aspects of biological microscopy, including the classification of macromolecules (Van Heel, 1984; Van Heel, 1989; Frank, 1990; Marabini and Carazo, 1994; Pascual, et al., 2000) and the segmentation of boundaries of cultured cells (Wu et al., 1996), it had yet to be applied to the segmentation of structures from serial EM datasets, and certainly not on the scale of what would be required to reconstruct even partial subcellular models or wiring diagrams of the brain.

Machine learning approaches developed for biological image segmentation fall into one of two categories: supervised or unsupervised. For the former, the user must supply the algorithm with a set of training images and labels that dictates how the different classes should be discriminated. If, for example, a membrane segmentation is desired, the user might supply the raw EM image as well as a binary image of the same size that partitions each pixel into the class "membrane" or "not membrane." If the algorithm supports it, additional labels may be added as desired, e.g. "membrane," "mitochondrion," and "background." In the case of unsupervised learning, no such training sets are required,

and the algorithm performs the classification solely based on information present in the image that is to be segmented. The major advantage of unsupervised algorithms is that they do not require the generation of training data, which must be performed by a human and, ideally, should be painstakingly accurate. The generation of such training data may take anywhere from a few hours to weeks, depending upon the needs of the given algorithm and the feature for which training is desired. However, supervised methods have consistently outperformed their unsupervised brethren when applied to complicated EM images (Tasdizen, et al., 2014). Since the correction of automatic segmentations, even those generated by the highest performing algorithms available, still requires considerable human effort (Plaza et al., 2014), the better performing supervised approaches remain the gold standards of the field.

A number of open-source software packages that perform supervised pixel classification for EM data have been made available to the public in recent years. The Fiji distribution of ImageJ (Schindelin et al., 2012) contains a “Trainable Weka Segmentation” plugin that uses the WEKA machine learning libraries (Hall et al., 2009) to train a Random Forest pixel classifier on a host of selectable image features. The plugin provides a GUI by which the user can create a training set by manually tracing lines corresponding to different classes on the input image. The trained model can be saved and intuitively applied to classify naïve test datasets. Though fairly user-friendly, this plugin for pixel classification has not seen widespread adoption by the biological sciences, in part because it requires a significant amount of time and memory to generate its classifiers.

While other open-source software packages for pixel classification, such as CellProfiler (Carpenter, et al., 2006) and BIOCAT (Zhou et al., 2013), have been introduced, it is the contribution of Fred Hamprecht’s group, ilastik (Sommer et al., 2011), that has seen the most extensive use in the 3D EM community. Similar to Fiji’s WEKA

plugin, ilastik provides a GUI that allows the user to interactively add labels to a training set. The user then selects the features and scales they wish to utilize for the training of a Random Forest classifier, and the generated voxel classifications for all classes are shown in real-time as they are generated. The classifier can then be stored to disk in the HDF5 format (Folk et al., 1999) and applied to test images in a batch mode. Alternatively, the user can operate in the so-called “headless mode”, where training and voxel classification are performed through the command line using previously generated training sets. Applications of ilastik to 3D EM datasets have been numerous, and include the segmentation of membranes (Andres et al., 2012), synapses (Kreshuk et al., 2014; Kreshuk et al., 2011), and cell nuclei (Tek et al., 2014). One caveat of ilastik is that its classifiers are trained on 3D features; it employs voxel classification rather than pixel classification. This is of minor concern for isotropic imaging modalities such as FIBSEM, but may cause issues for anisotropic modalities such as SBEM and ssTEM. First, SBEM or ssTEM image stacks may need to be downsampled, often dramatically, in-plane to achieve near-isotropic voxels. Second, 3D classification leaves the results susceptible to any number of errors that may occur during imaging and processing, including surface charging, focal gradients, specimen overlap, and imperfectly aligned sections.

With relatively few exceptions, most machine learning approaches for organelle segmentation have focused on mitochondria (Vitaladevuni et al., 2008; Narashima et al., 2009; Smith et al., 2009; Kumar et al., 2010; Seyedhosseini et al., 2013a). Recently, Giuly and co-workers proposed a method to segment mitochondria utilizing patch classification followed by isocontour pair classification and level sets (Giuly, et al., 2012). Lucchi and colleagues (Lucchi, et al., 2012; Lucchi et al., 2010) developed an approach that trains a classifier to detect supervoxels that are most likely to belong to the boundary of the desired organelle. A method to automatically segment cell nuclei using ilastik to train a Random

forest voxel classifier followed by morphological post-processing and object classification was proposed by Tek and colleagues (Tek, et al., 2014).

One drawback of these proposed methods is that they make critical assumptions about the geometry of their segmentation target that render their expansion to other organelles nontrivial. The workflow developed by Tek and colleagues to produce nuclear segmentations, for example, would not be readily applicable to the segmentation of mitochondria without heavy modifications to its filters. Additionally, many published methods have only been tested on very small subregions of datasets. The method of Giuly and colleagues, for example, was only tested on a 350 x 350 x 30 voxel slab of data (Giuly et al., 2012). Since most large-scale 3D EM datasets contain a heterogeneous mixture of organelle morphologies spread across different cell types, it is unclear if such methods would achieve uniform results when applied to such datasets. Astrocytic mitochondria, for example, are known to be more elongated than their neuronal counterparts (Fernandez et al., 1983). Filters based on geometry and morphology may therefore preferentially accept mitochondria from one cell type and reject those from another.

Taking the above into consideration, it is clear that there remains a need for a method to accurately segment various organelles in SBEM stacks without any *a priori* assumptions about organelle morphology. In this chapter, a method for the robust and accurate automatic segmentation of morphologically and functionally diverse organelles in EM image stacks is presented. Organelle-specific pixel classifiers are trained using the cascaded hierarchical model (CHM; Seyedhosseini et al., 2013b), a state-of-the-art, supervised, multi-resolution framework for image segmentation that utilizes only 2D image information. Critically, since only 2D features are considered, this method is equally applicable to both isotropic and anisotropic imaging modalities. A series of tunable 2D filters are then applied to generate accurate segmentations from the outputs of pixel

classification. In the final processing step, 3D connected components are meshed together in a manner that minimizes the deleterious effects of local and global imaging artifacts. Finally, an efficient workflow for scaling this method to teravoxel-sized datasets that leverages upon parallelization with supercomputing resources is presented.

## **2.2. Methods Development and Results**

In this section, an algorithm developed for the automatic segmentation of multi-scale features in SBEM image stacks will be described. The steps of the algorithm, which utilizes CHM for pixel classification, will be outlined in detail, and performance metrics will be presented for a variety of sub-cellular organelles (mitochondria, lysosomes, nuclei, and nucleoli) using a single SBEM image stack as the test dataset. The strength of this algorithm will then be established by means of a comparison to a recently reported method for the automatic segmentation of nuclei in SBEM datasets (Tek et al., 2014).

In the first sub-section, the parameters used for imaging and pre-processing of the dataset will be defined. Scripts for expediting image downsampling, conversion, and histogram modification will be introduced. In the second sub-section, the generation of organelle-specific CHM pixel classifiers will be outlined, and the description of a computationally feasible workflow for the application of these classifiers to teravoxel-sized image stacks will follow. A novel method for the binarization of CHM probability maps will then be introduced, and its performance will be compared against that of other published methods.

### **2.2.1. Data Collection and Pre-processing**

#### **2.2.1.1. Tissue processing and SBEM image stack acquisition**

The suprachiasmatic nucleus (SCN) of one three-month-old, male C57BL/6J mouse was harvested and prepared for SBEM using a standard protocol (Wilke, et al., 2013). The resin-embedded tissue was mounted on an aluminum specimen pin and prepared for SBEM imaging as previously described (Holcomb, et al., 2013). Imaging was performed by detection of backscattered electrons (BSE) using a Zeiss Merlin scanning electron microscope equipped with a 3View ultramicrotome (Gatan). The SBEM image stack was acquired in ultrahigh vacuum mode using an accelerating voltage of 1.9 kV, a pixel dwell time of 500 ns, and a spot size of 1.0. Sectioning was performed with a cutting thickness of 30 nm. BSE images were acquired at 800x magnification with a raster size of 32,000 pixels x 24,000 pixels, yielding a pixel size of 3.899 nm/pixel. A total of 1,283 serial images were acquired, resulting in an image stack with tissue dimensions of roughly 124.8  $\mu\text{m}$  x 93.6  $\mu\text{m}$  x 38.5  $\mu\text{m}$  ( $\sim 450,000 \mu\text{m}^3$ ). The specimen was then removed from the chamber, and an image of a diffraction grating replica specimen (Ted Pella, Redding, CA, U.S.A.) was acquired at the same magnification for calibration of the lateral pixel size. Low magnification images of the block-face were acquired before and after sectioning.

### **2.2.1.2. SBEM stack alignment**

All individual images of the input SBEM stack were converted to the MRC format and appended to an 8-bit MRC stack using the IMOD programs *dm2mrc* and *newstack*, respectively (Kremer, et al., 1996). Sequential images within the stack were then translationally aligned to one another in the XY-plane using the cross-correlational alignment algorithm of the IMOD program *tiltxcorr*.

The lateral pixel size of the stack was determined using the calibration image of the diffraction grating replica specimen. The calibration image was opened in 3dmod, and an object containing twenty open contours was initialized. On each contour, a line covering

ten grid boxes was traced, and the pixel lengths of each open contour were computed using the IMOD program *imodinfo*. Each length was divided by ten to give the number of pixels per grid box, and subsequently converted to nanometers using the known size of each grid box. The average of these lengths for all twenty contours was determined, and this value was used as the final lateral pixel size of the stack. The header information of the aligned MRC stack was then edited to reflect the true pixel size using the IMOD program *alterheader*.

### **2.2.1.3. Image downsampling and conversion**

Since working with a 32,000 x 24,000 pixel image size is unwieldy even for modern computers, the ability to laterally downsample images in the stack without sacrificing the accuracy of automatically generated segmentations is highly beneficial. Downsampling by only a factor of two would reduce the single image size to 16,000 x 12,000 pixels, a total size reduction of four times. However, the allowable degree of downsampling is likely to vary depending on the segmentation target; larger targets, such as nuclei, may allow for higher levels of downsampling before an impact on segmentation accuracy is noticeable. In order to test this, laterally downsampled versions of the same dataset were generated with binning factors of 2, 4, 6, 8, and 10. Additionally, a version with isotropic voxels was generated by laterally downsampling by a factor of 7.694, the ratio of the axial resolution to the lateral pixel size.

To allow for easy and efficient downsampling, *newstack\_bin.sh*, a Bash script that incorporates IMOD programs, was written (Appendix C.2.1). The script requires three inputs: (1) the location of the original MRC stack, (2) the factor by which it should be downsampled, and (3) the name for the downsampled MRC stack. Each slice in the

**Table 2.1. An expedited approach to the downsampling of SBEM image stacks.** All slices from an SBEM stack were individually extracted and downsampled in a two-step process. The number of pixels per slice is indicated at each level of downsampling. The average times for slice extraction and downsampling are reported as the mean  $\pm$  standard deviation. Following this process, all images were appended to a final, downsampled stack; the time required for this stacking is also indicated. Though extraction time is independent of image size, both downsampling and stacking times decrease as the size of the input image decreases.

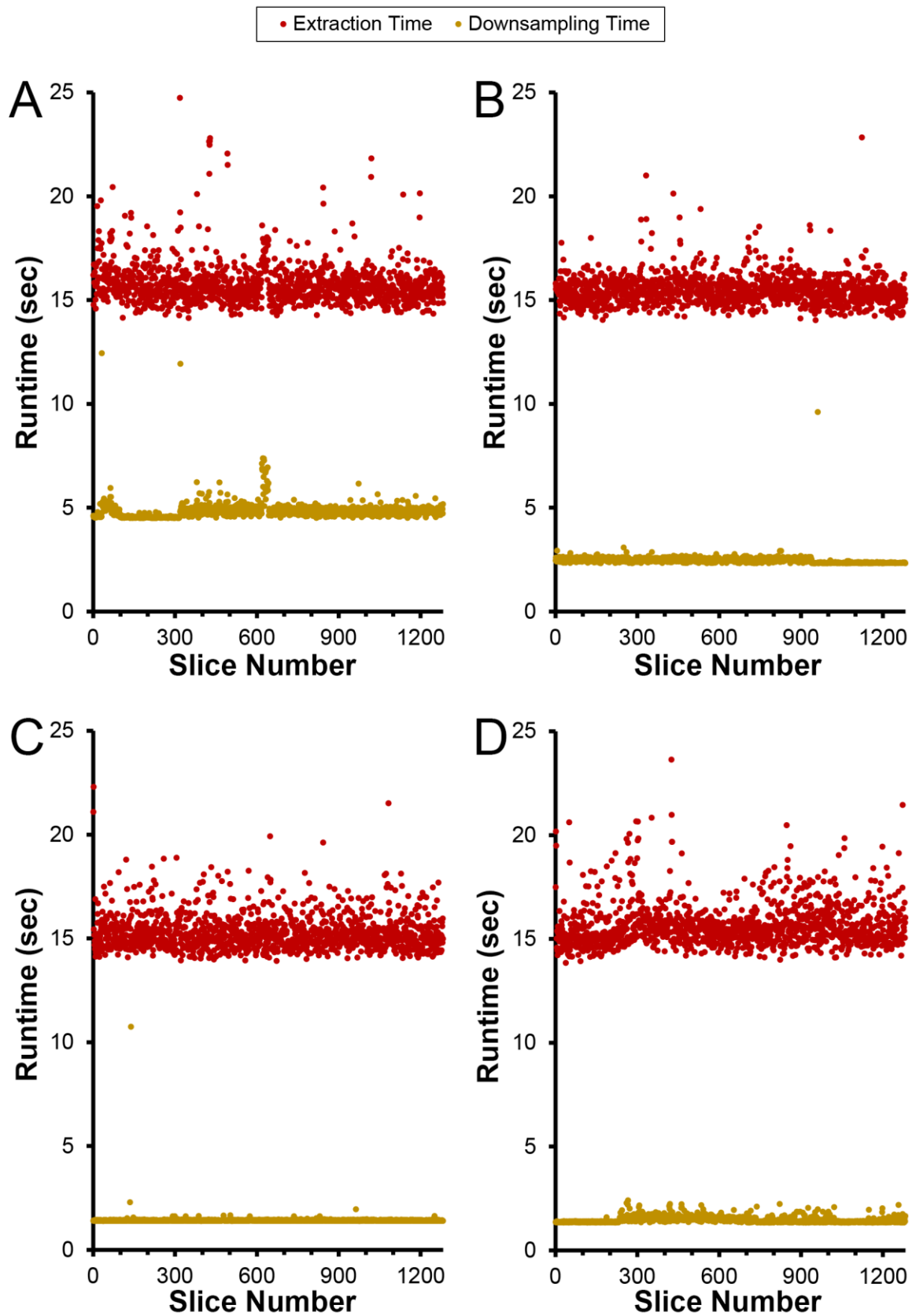
Binning Factor	Number of Pixels	Average Extraction Time Per Slice (sec.)	Average Downsampling Time Per Slice (sec.)	Stacking Time (min.)
1	$7.68 \times 10^8$	-	-	-
2	$1.92 \times 10^8$	$15.77 \pm 1.20$	$4.85 \pm 0.43$	52.35
4	$4.80 \times 10^7$	$15.46 \pm 0.72$	$2.48 \pm 0.74$	13.23
8	$1.20 \times 10^7$	$15.29 \pm 0.87$	$1.42 \pm 0.26$	4.09
10	$7.68 \times 10^6$	$15.64 \pm 1.06$	$1.46 \pm 0.15$	2.69

input stack is extracted to a temporary, single-image MRC file and then downsampled by the user-specified factor using the IMOD program *newstack*. Afterwards, all binned, single-image MRC files are appended to a new MRC stack with the user-specified output name. By downsampling only single images, this approach eliminates the risk of stalling that may occur when using the entire stack as input to the *newstack* program. The number of pixels, average time per slice for extraction and downsampling, and the total time required for stacking are reported in Table 2.1. Plots indicating the time required for extraction and downsampling for every slice in the stack are shown in Figure 2.6. The results shown in Figure 2.6 demonstrate that downsampling using the script *newstack\_bin.sh* runs smoothly without any stalling for the entirety of the stack.

The code for pixel classification with CHM requires test images to be in the Portable Network Graphics (PNG) format. Each downsampled MRC stack was converted to a set of sequentially numbered PNG files using the Bash script *mrcstack2png.sh* (Appendix C.2.2). This script requires two inputs: (1) the location of the MRC stack and (2) the path to store output PNG files to. An array job is then submitted using the Sun Grid



**Figure 2.6. The proposed method for image downsampling processes entire datasets without stalling.** The times required to extract and downsample each slice from the same native resolution SBEM dataset are shown at different levels of downsampling (A, 2x downsampling; B, 4x downsampling; C, 8x downsampling; D, 10x downsampling). The proposed method runs smoothly for all images without demonstrating any significant spikes in the time required. Extraction times are, as expected, relatively consistent and independent of the desired level of downsampling. Downsampling times roughly scale with the desired level of downsampling over the range of 2x – 8x. At 10x downsampling, the average downsampling time per slice is virtually the same as for 8x (C,D).



Engine (SGE) job script *mrcstack2png.q* (Appendix C.2.3). This allows slices to be processed in parallel and expedites the conversion process.

#### **2.2.1.4. Histogram Equalization**

The histograms of all images in the stack were equalized using a MATLAB (The MathWorks, Inc., Natick, MA, U.S.A.) implementation of the exact histogram specification (EHS) algorithm (Coltuc et al., 2006). Unlike global histogram equalization (GHE), which attempts to improve contrast by assigning equal numbers of pixels to the intensity bins of the image, histogram specification attempts to fit the image's histogram to a given function or spread. The EHS algorithm is a type of histogram specification that allows for the modification of an image's histogram to fit any desired histogram with discrete bins. Therefore, the first step is to generate a reference histogram that all images in the stack will be normalized to. Two approaches for generating this reference histogram are considered: (1) the use of the histogram of a single slice, and (2) the use of the summed histogram of the entire stack.

Irrespective of which approach is chosen, artifacts introduced by image borders must be eliminated before an accurate histogram for specification can be extracted. When lateral translations are needed to align successive slices to one another via cross-correlation, a border of uniform pixel intensity is formed around the image to maintain the same image dimensions following translation (Figure 2.7, A-B). Such a border causes a spike in the histogram at this pixel intensity (Figure 2.7, D). Using simple translational alignments, borders may exist on as many as two edges and as few as zero. If more advanced alignments accounting for warping are implemented, borders can exist on all four edges and may result in a sheared image with non-uniform borders. Therefore, an

automatic method for detecting borders that could be applied to an image aligned with any method was necessary.

The implemented method for border detection calculates the magnitude of the numerical pixel gradient of the input image. Since borders have the same pixel intensity throughout, the gradient magnitude of pixels belonging to borders will be approximately zero. Gradients are computed using a MATLAB script, *find\_nonborder\_pixels.m* (Appendix C.2.4), which calculates the horizontal ( $G_x$ ) and vertical ( $G_y$ ) gradients of the input image ( $I$ ) of size  $M \times N$  pixels according to the following formulae:

$$G_{x_{i,j}} = \begin{cases} I_{2,j} - I_{1,j}, & i = 1 \\ \frac{1}{2}(I_{i+1,j} - I_{i-1,j}), & 2 \leq i \leq M - 1 \\ I_{M,j} - I_{M-1,j}, & i = M \end{cases} \quad \forall j \in \{1, \dots, N\}$$

$$G_{y_{i,j}} = \begin{cases} I_{i,2} - I_{i,1}, & j = 1 \\ \frac{1}{2}(I_{i,j+1} - I_{i,j-1}), & 2 \leq j \leq N - 1 \\ I_{i,N} - I_{i,N-1}, & j = N \end{cases} \quad \forall i \in \{1, \dots, M\}$$

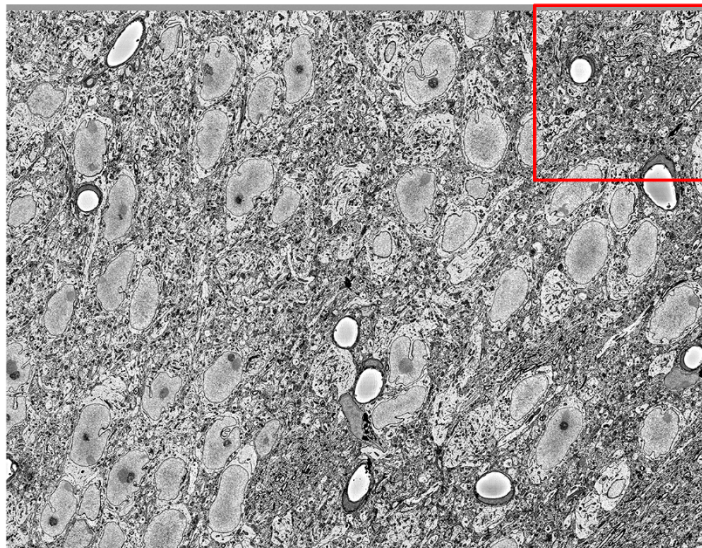
The gradient magnitude is then calculated for each pixel:

$$|G|_{i,j} = \sqrt{G_{x_{i,j}}^2 + G_{y_{i,j}}^2}$$

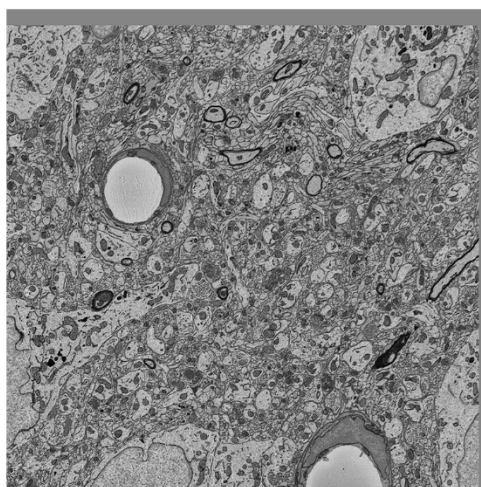
This gradient magnitude image is binarized using a single-level threshold at a low value ( $\sim 0.01$ ) to automatically detect border pixels. Occasionally, false positives may be detected on the interior of the image; such false positives typically occur at blood vessels, whose lumens may also have very small pixel gradients. To eliminate such false positives, the mask obtained from thresholding is inverted, and 2D hole-filling is performed. Thus, the final binary mask output from *find\_nonborder\_pixels.m* has all image pixels marked as positive and all border pixels marked as negative. An example of this mask is shown in Figure 2.7C, which depicts the automatically detected non-border pixels with a transparent green overlay. The histogram of the whole masked image slice is shown in Figure 2.7E,

**Figure 2.7. The automatic detection of image borders.** As a result of various alignment algorithms, SBEM image slices often possess borders of uniform pixel intensity (A, boxed region magnified in B). Such borders add a spike artifact to the image histogram (C) that is not representative of the true data in the image. By computing the magnitude of the image gradient to detect non-border pixels, image pixels can be reliably discriminated from border pixels (D). The histogram of only the true image pixels can then be determined; such a histogram is shown to be devoid of the spike artifact (E) and is acceptable for use in histogram specification algorithms.

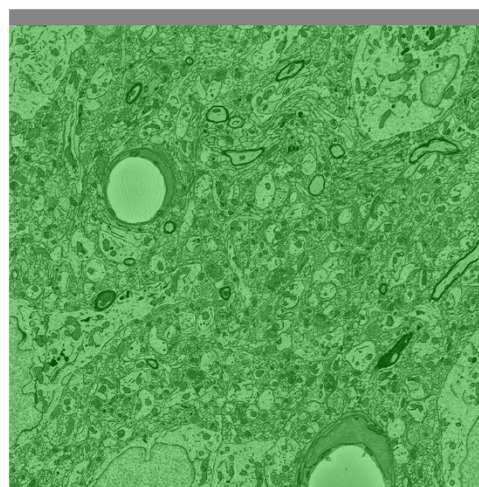
A



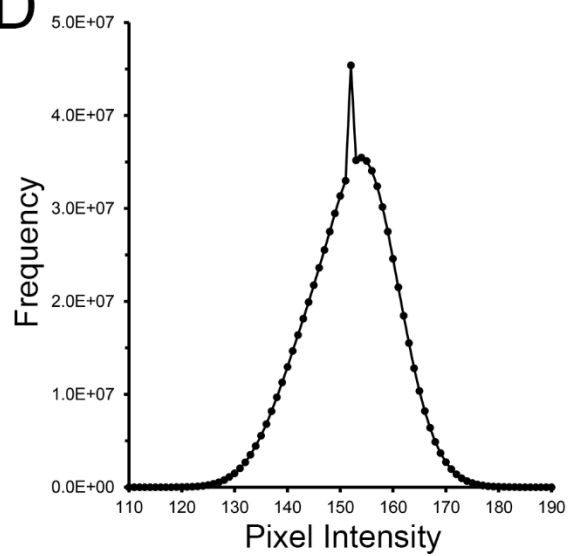
B



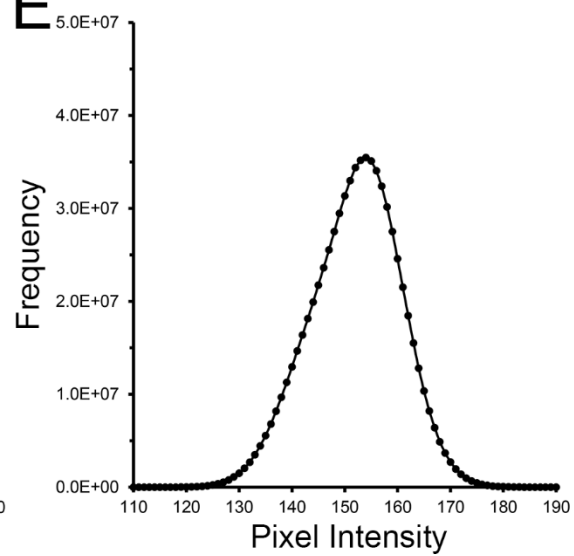
C



D



E



which displays the same shape of the histogram in Figure 2.7D, but without the spike in pixel intensity due to border pixels.

Reference histogram generation is performed by invoking the Bash wrapper script *generate\_reference.sh* (Appendix C.2.5), which requires three inputs: (1) the path containing the stack of PNG images generated in Section 2.2.1.3, (2) an output path for reference histograms, and (3) the desired mode of operation. If single-slice mode is chosen, the desired slice number for the reference image must be specified. In this case, the SGE job script *generate\_reference.q* (Appendix C.2.6) is submitted, which invokes the MATLAB function *generate\_reference.m* (Appendix C.2.7) to mask non-border pixels of the input slice as previously described. The 8-bit histogram of these non-border pixels is then computed and written to an ASCII file with 256 lines, in which each line corresponds to the pixel count at the given pixel intensity. If full-stack mode is chosen, *generate\_reference.q* is submitted as an array job such that ASCII files specifying the histogram of every image in the stack are output.

Once reference histograms have been generated, histogram equalization is performed using a MATLAB implementation of the EHS algorithm downloaded from the MATLAB File Exchange (File ID: #26309). EHS processing is initialized using the Bash wrapper script *run\_ehs.sh* (Appendix C.2.8), which requires three inputs: (1) the path to the stack of PNG images, (2) the path to the reference histogram ASCII files, and (3) an output path to store equalized PNG images to. This wrapper script submits an SGE array job using the job script *run\_ehs.q* (Appendix C.2.9), such that images can be processed in parallel. Each job invokes the MATLAB function *run\_ehs.m* (Appendix C.2.10), which first loads and sums all reference histograms in the specified path to yield the final full-stack reference histogram. If single-slice mode was used, the single reference histogram will be loaded without summation. EHS is then applied to the non-border pixels of the

image using the summed reference histogram as the target. The conclusion of this process yields a stack of PNG images whose overall histograms are all essentially identical to one another.

## 2.2.2. Pixel classification

### 2.2.2.1. Generation of training labels

For each organelle target, a set of training images and labels was generated. First, a set of 50 seed points,  $P_i$ , were selected for each organelle throughout the processed SBEM stack such that:

$$P_i = (x, y, z) \forall i \in \{1, \dots, 50\}$$

These points were chosen in a manner that yielded a wide distribution throughout the stack. Since subtle alterations in image quality may occur throughout an SBEM dataset, this wide distribution was preferable to simply taking training images from the same region of consecutive slices, as such a sub-volume may not be representative of the whole dataset.

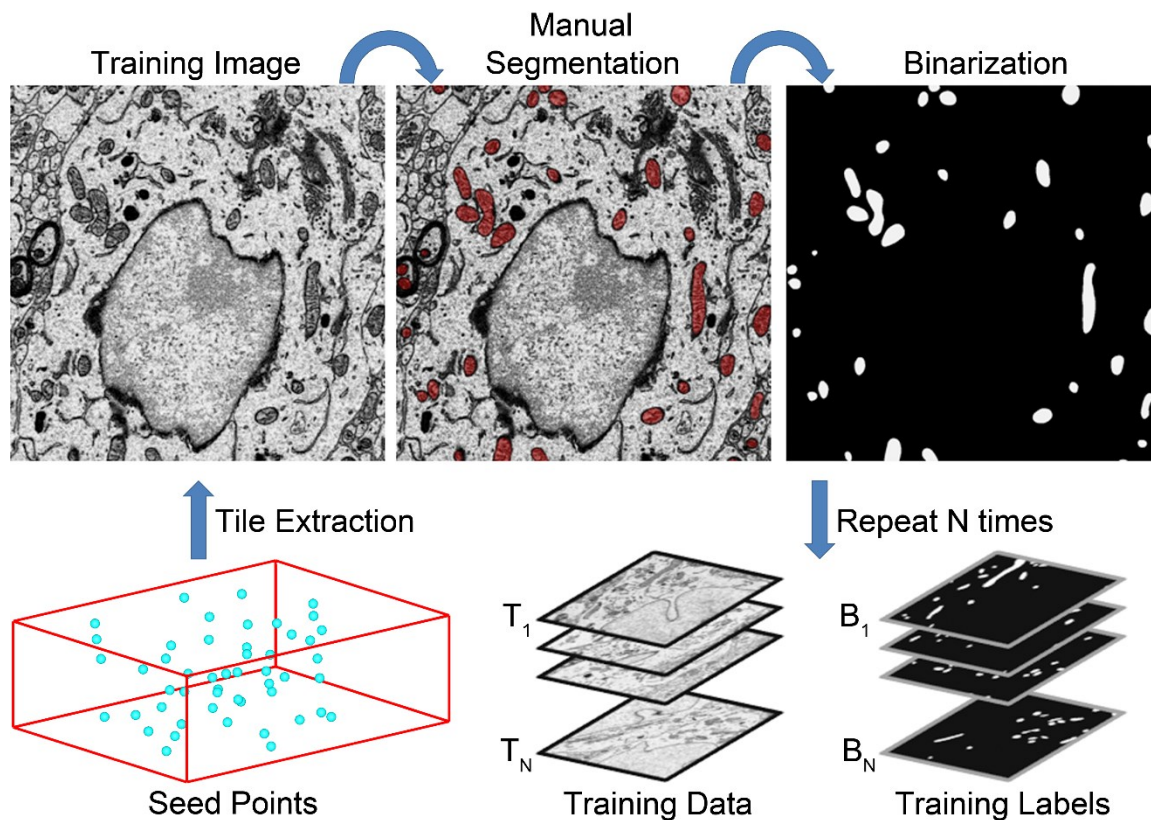
After the selection of seed points, every instance of the chosen organelle was manually segmented in a 500 x 500 pixel tile centered at each  $P_i$ . To maintain consistency, the manually segmented contours were placed on the inside of the membrane of membrane-bound organelles. Once segmentations were completed, training images and labels were extracted using the scripts *process\_td.sh* (Appendix C.2.11) and *process\_td.q* (Appendix C.2.12). These scripts work by first extracting the coordinates of the points  $P_i$  to a text file using the IMOD program *model2point*. Then, using the IMOD program *trimvol*, 2D tiles of size 500 x 500 pixels are extracted, masked using *imodauto* with the manually generated contours, and converted to PNGs with the IMOD program *mrc2tif*. These masked images are binary representations of the organelle of interest, and serve as



training labels. Training images are then extracted from the corresponding full-size images in the EHS PNG stack using ImageMagick's *convert* program. Thus, the final outputs from training data generation are (1) a stack of 8-bit, grayscale training images,  $T_i$ , and (2) a stack of corresponding binary organelle masks,  $B_i$ . A flow chart illustrating this procedure for training data generation is shown in Figure 2.8. Training sets were generated for four organelles of interest: nuclei, nucleoli, mitochondria, and lysosomes. The training images and labels used for training each classifier are shown in Appendix B. Additional training sets were also generated for the downsampled image stacks created in Section 2.2.1.3 using the same manual segmentations. Thus, for each organelle, training sets were created for levels of downsampling of 2, 4, 6, 8, and 10.

#### **2.2.2.2. Training organelle-specific classifiers**

The CHM consists of bottom-up and top-down steps cascaded in multiple stages (Seyedhosseini, et al., 2013b). The bottom-up step occurs in a user-specified number of hierarchical levels,  $L$ . At each level, the input stacks  $T_i$  and  $B_i$  are sequentially downsampled and a classifier is trained based on features extracted from the downsampled data as well as information from all lower levels of the hierarchy. After classifiers have been trained at all levels, the top-down path combines the coarse contextual information from higher levels into a single classifier that is applicable to images at native resolution. This whole process is then cascaded in a number of stages,  $S$ , where the output classifier from the previous stage serves as the input classifier for the subsequent stage. The final output is a pixel classifier,  $C_{S,L}$ , that is applicable to images at the native pixel size of  $T_i$  and  $B_i$ . For optimal results, the number of stages chosen should be greater than one. The exact number of stages and levels chosen depends on a host of factors, including the size of  $T_i$  and  $B_i$  and the computational resources available to the



**Figure 2.8. A flow chart of the steps involved in training data generation.** The generation of a set of training data for mitochondrial automatic segmentation is shown here. First, a set of seed points,  $P_i$ , were selected such that a wide distribution throughout the volume is achieved (bottom left). Tiles of size  $500 \times 500$  centered at each seed point were extracted to serve as training images,  $T_i$ . All instances of the desired organelle target were manually segmented on each training image. These manual segmentations were then used as masks to binarize each  $T_i$  such that pixels of value one correspond to pixels of  $T_i$  that are positive for the desired organelle. This process was repeated 50 times to yield stacks of training images and their corresponding training labels,  $B_i$ .

experimenter.

For each organelle target, 90 seed points were placed throughout the SBEM stack as described in Section 2.2.2.1. Of the 90 tiles generated for each organelle, 50 were randomly selected for use in training a CHM classifier; the other 40 were set aside to use as test data and ground truth for evaluating classifier performance. CHM classifiers were trained with two stages and two levels for each target organelle, and at each level of downsampling. The wallclock time and memory requirements for training each classifier are given in Table 2.2. All classifiers were trained using the high memory node (monster-2.8) of the National Biomedical Computation Resource (NBCR) cluster, rocce.ucsd.edu (<http://rocce-mgr.ucsd.edu/>).

### **2.2.2.3. Computation of probability maps**

For each organelle and level of downsampling, the corresponding set of 40 test tiles were subjected to pixel classification using the appropriate trained classifier. Pixel classification was performed in parallel by submitting the SGE array job script *CHM\_array\_testTile.q*. An example of this script is shown in Appendix C.2.13. A tiling routine was built and incorporated into the script for cases in which the test images are significantly larger than the data used to train the classifier. When desired, the user can specify a desired number of tiles in the X and Y directions. For example, if the test image has dimensions of 4,000 pixels x 2,000 pixels and the classifier was trained on images of size 500 x 500, tiling with dimensions of 8 x 4 would be appropriate. Each input image is then decomposed into the specified number of tiles using a routine built around a series of ImageMagick commands (Appendix C.2.13). The user is also able to specify the number of pixels by which adjacent tiles should overlap. Probability maps are then computed for each tile. Once all tiles have been generated, they are automatically stitched

**Table 2.2. Computational requirements for organelle-specific pixel classification.**

The dimensions of the stack of training images and labels used to train the classifier are given. The values for pixel classification correspond to the average values required to generate a probability map for one tile of roughly 60  $\mu\text{m}^2$  at the tissue level (1,000 x 1,000 pixels at 2x downsampling). Values are reported as the mean and standard deviation (N = 40 for each). Time is reported as the wall clock time for the indicated process.

nm/pixel	Classifier Training			Pixel Classification	
	Dimensions	Time (hr.)	RAM (GB)	Time (min.)	RAM (GB)
<b>Mitochondria</b>					
7.79	500 x 500 x 50	22.27	78.54	13.25 ± 1.18	4.54 ± 0.04
15.59	250 x 250 x 50	17.69	39.40	4.66 ± 2.07	2.08 ± 0.05
23.39	166 x 166 x 50	7.74	18.09	2.07 ± 0.06	1.69 ± 0.05
31.19	125 x 125 x 50	2.68	10.77	1.17 ± 0.03	1.49 ± 0.08
38.90	100 x 100 x 50	2.59	7.31	0.93 ± 0.05	1.40 ± 0.06
<b>Lysosomes</b>					
7.79	500 x 500 x 50	43.45	75.78	13.12 ± 0.61	4.53 ± 0.03
15.59	250 x 250 x 50	38.39	39.34	4.71 ± 0.18	2.09 ± 0.06
23.39	166 x 166 x 50	12.13	18.06	2.06 ± 0.05	1.68 ± 0.05
31.19	125 x 125 x 50	6.65	10.78	1.11 ± 0.01	1.51 ± 0.04
38.90	100 x 100 x 50	4.08	7.35	0.82 ± 0.02	1.47 ± 0.04
<b>Nuclei</b>					
7.79	500 x 500 x 50	23.98	87.24	12.73 ± 0.90	4.54 ± 0.03
15.59	250 x 250 x 50	20.35	39.38	4.67 ± 0.15	2.08 ± 0.04
23.39	166 x 166 x 50	7.95	18.16	2.03 ± 0.03	1.68 ± 0.05
31.19	125 x 125 x 50	4.71	10.83	1.18 ± 0.02	1.52 ± 0.04
38.90	100 x 100 x 50	3.18	7.38	0.90 ± 0.04	1.41 ± 0.04
<b>Nucleoli</b>					
7.79	500 x 500 x 50	20.67	81.80	13.56 ± 1.70	4.54 ± 0.03
15.59	250 x 250 x 50	22.67	39.35	4.76 ± 0.16	2.09 ± 0.04
23.39	166 x 166 x 50	10.10	18.06	2.06 ± 0.04	1.69 ± 0.04
31.19	125 x 125 x 50	5.75	10.84	1.17 ± 0.02	1.50 ± 0.04
38.90	100 x 100 x 50	3.06	7.34	0.89 ± 0.02	1.41 ± 0.04

together to yield the final, full-sized probability map of the complete image. During stitching, regions of overlap are handled by taking either the average or maximum pixel intensity across the overlapped regions from all appropriate tiles. Following stitching, the final probability map is normalized such that each pixel ranges from  $[0, \dots, 1]$ , with one representing the greatest probability of a true positive.

#### 2.2.2.4. Assessment of classifier performance

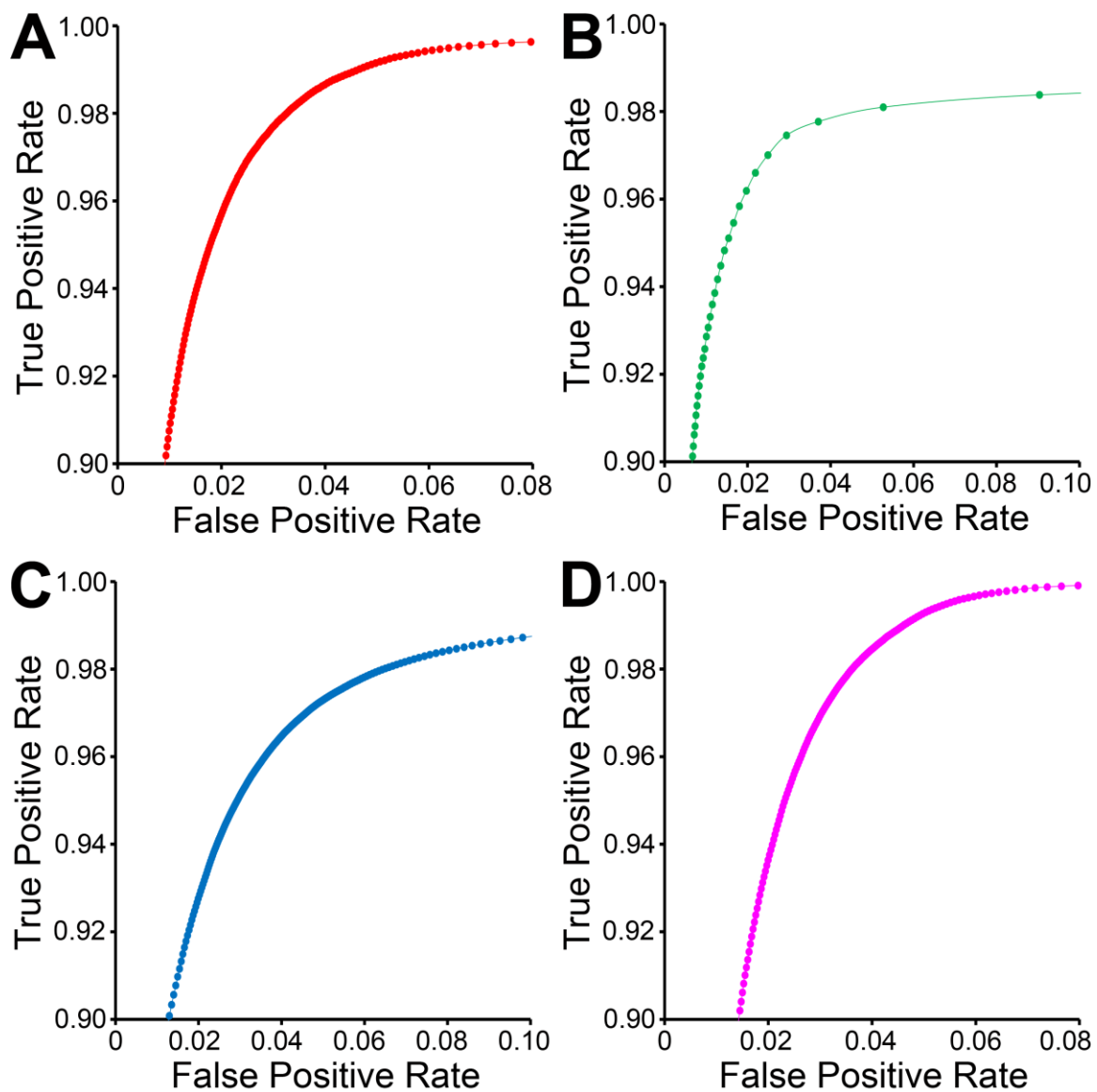
Following pixel classification of all test images, classifier performance was assessed by comparing the output, normalized probability maps to their corresponding, manually segmented ground truth. Receiver operating characteristic (ROC) and precision-recall curves (Fawcett, 2006) were generated by applying pixel intensity thresholds ranging from 0 to 1 in increments of 0.01 to each probability map. The confusion matrix was computed at each threshold value, and the true positive rate (TPR, or recall), false positive rate (FPR), and precision were calculated according to the following formulae:

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

ROC curves were generated by plotting TPR against FPR for all threshold values, and precision-recall curves were generated by plotting precision against TPR for all threshold values. The computation of these values was performed using the MATLAB script *segStats.m* (Appendix C.2.14). ROC curves for the four organelle targets at a downsampling level of 2x are shown in Figure 2.9. In addition to the values described above, the script *segStats.m* also calculates a host of other segmentation evaluation



**Figure 2.9. ROC curves for CHM classifiers of various organelles.** ROC curves for mitochondrial (A), lysosomal (B), nuclear (C), and nucleolar (D) CHM classifiers generated with two stages and two levels.

metrics, including the true negative rate (TNR), false negative rate (FNR), false discovery rate (FDR), negative predictive value (NPV), accuracy, F-value, Jaccard coefficient (Powers, 2011; Lucchi et al., 2012), geometric mean (Seyedhosseini et al., 2013b), and Matthew's correlation coefficient (Matthews, 1975; Baldi et al., 2000). These metrics are calculated according to the following formulae:

$$\text{TNR} = \frac{\text{TN}}{\text{FP} + \text{TN}}$$

$$\text{FNR} = \frac{\text{FN}}{\text{FN} + \text{TP}}$$

$$\text{NPV} = \frac{\text{TN}}{\text{TN} + \text{FN}}$$

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{FP} + \text{TN}}$$

$$\text{Jaccard} = \frac{\text{TP}}{\text{FP} + \text{TP} + \text{FN}}$$

$$\text{G-mean} = \sqrt{\text{TPR} \times \text{TNR}}$$

$$\text{MCC} = \frac{\text{TP} \times \text{TN} - \text{FP} \times \text{FN}}{\sqrt{(\text{TP} + \text{FP})(\text{TP} + \text{FN})(\text{TN} + \text{FP})(\text{TN} + \text{FN})}}$$

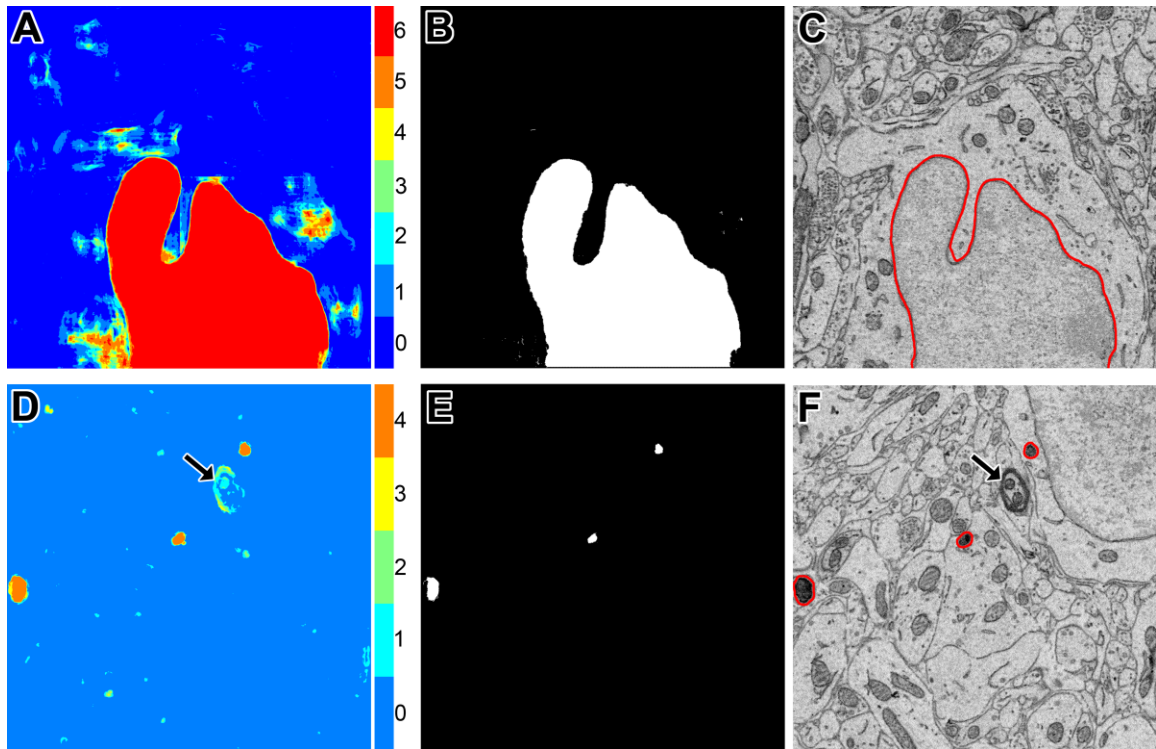
#### 2.2.2.5. Binarization of probability maps

Each probability map,  $M_j$ , is binarized by evolving active contours (Chan and Vese, 2001) at automatically determined initial positions. For an unsupervised determination of the initial positions, the probability map  $M$  is first thresholded using Otsu's multi-level method (Otsu, 1979) with  $G$  unique gray levels. The output from this operation is  $O_j$ , a map in which each pixel of  $M_j$  has been classified into one of  $G$  unique levels, with the zeroth level corresponding to the approximate background. This map is then binarized by thresholding  $O_j$  at a pixel intensity of  $G$ , yielding a mask of initial positions,  $K_j$ . This binary mask is then made smaller by applying two iterations of morphological shrinking and used

to initialize the evolution of active contours with a number of iterations and smoothing factor specified by  $\alpha$  and  $\lambda$ , respectively. Each 2D connected component of  $K_j$  serves as a unique initial position for contour evolution. For best results,  $\alpha$  should be at least 50. The choice of  $\lambda$  depends largely on the organelle target and pixel size of the test images, but in general should fall in the range of 0-8. Larger values of  $\lambda$  can be used when the pixel size is small. If the pixel size is too large (i.e. above 10 nm/pixel), smoothing should be turned off by setting  $\lambda$  to zero. The value of  $G$  significantly alters the results, and its choice is dependent on the goals of the experimenter. Low values of  $G$  tend to emphasize true positives at the risk of retaining false positives. As  $G$  is increased, false positives are more readily removed, but so are true positives. The final output from this process is  $SEG_j$ , the organelle segmentation of the input grayscale image,  $I_j$ . An illustration of this process is shown for two test images in Figure 2.10. This binarization algorithm is implemented in the MATLAB script *binarize\_pm\_activecontour.m* (Appendix C.2.15).

The results of this method were compared to segmentations generated from the same probability maps, but with a number of different unsupervised binarization algorithms: (1) Minimum error thresholding (Kittler and Illingworth, 1986), (2) Maximum entropy thresholding (Kapur, et al., 1985), and (3) Otsu's single-level method (Otsu, 1979). The performance of each algorithm, as quantified by the F-value, Jaccard index, precision, and recall, was compared against that of the proposed method for each organelle target. The results of this comparison are shown in Table 2.3. The proposed active contour segmentation method resulted in a superior recall for all four organelles and a superior F-value for mitochondria, lysosomes, and nucleoli when compared to the other segmentation methods. The F-value for nuclear segmentation is negligibly better using Otsu's single-level method. The lack of distinction between these two binarization methods for nuclei is due largely to the already high quality of nuclear probability maps. The accuracy values





**Figure 2.10. The binarization of probability maps using active contours initialized by a multi-level Otsu threshold yields accurate segmentation results.** Colorized maps,  $M$ , of a nucleus (A) and lysosomes (D) generated by applying Otsu's method with multiple levels to probability maps obtained by CHM pixel classification. Each color corresponds to a unique level of the threshold. Six gray levels ( $G = 6$ ) were used for the nucleus and four ( $G = 4$ ) were used for the lysosomes. Initial positions (B, E) were determined by selecting pixels corresponding to only the highest levels of each threshold followed by two iterations of morphological shrinking. Output segmentations (C, F) were obtained by evolving active contours about each of the initial positions in (B) and (E) with 100 iterations and a smoothing factor of 8 ( $\alpha = 100, \lambda = 8$ ). In the case of the lysosome images, note that a myelinated axon that was originally detected by the classifier as a false positive (D, arrow) has been removed from the final segmentation by the application of our method (F, arrow).

**Table 2.3. Segmentation evaluation metrics for the tested organelle targets using various methods of probability map binarization.**

	F-value	Precision	Recall	Jaccard Index
<b>Mitochondria</b>				
Minimum Error	0.635	0.994	0.466	-
Max. Entropy	0.669	0.991	0.505	-
Otsu Single-level	0.816	0.957	0.712	-
Active Contours	0.877	0.867	0.886	0.780
<b>Lysosomes</b>				
Minimum Error	0.433	0.985	0.277	-
Max. Entropy	0.492	0.940	0.508	-
Otsu Single-level	0.812	0.899	0.737	-
Active Contours	0.841	0.854	0.828	0.726
<b>Nuclei</b>				
Minimum Error	0.963	0.958	0.968	-
Max. Entropy	0.644	0.603	0.692	-
Otsu Single-level	0.971	0.979	0.963	-
Active Contours	0.970	0.973	0.968	0.942
<b>Nucleoli</b>				
Minimum Error	0.781	0.998	0.641	-
Max. Entropy	0.811	0.996	0.684	-
Otsu Single-level	0.898	0.973	0.835	-
Active Contours	0.910	0.902	0.918	0.835

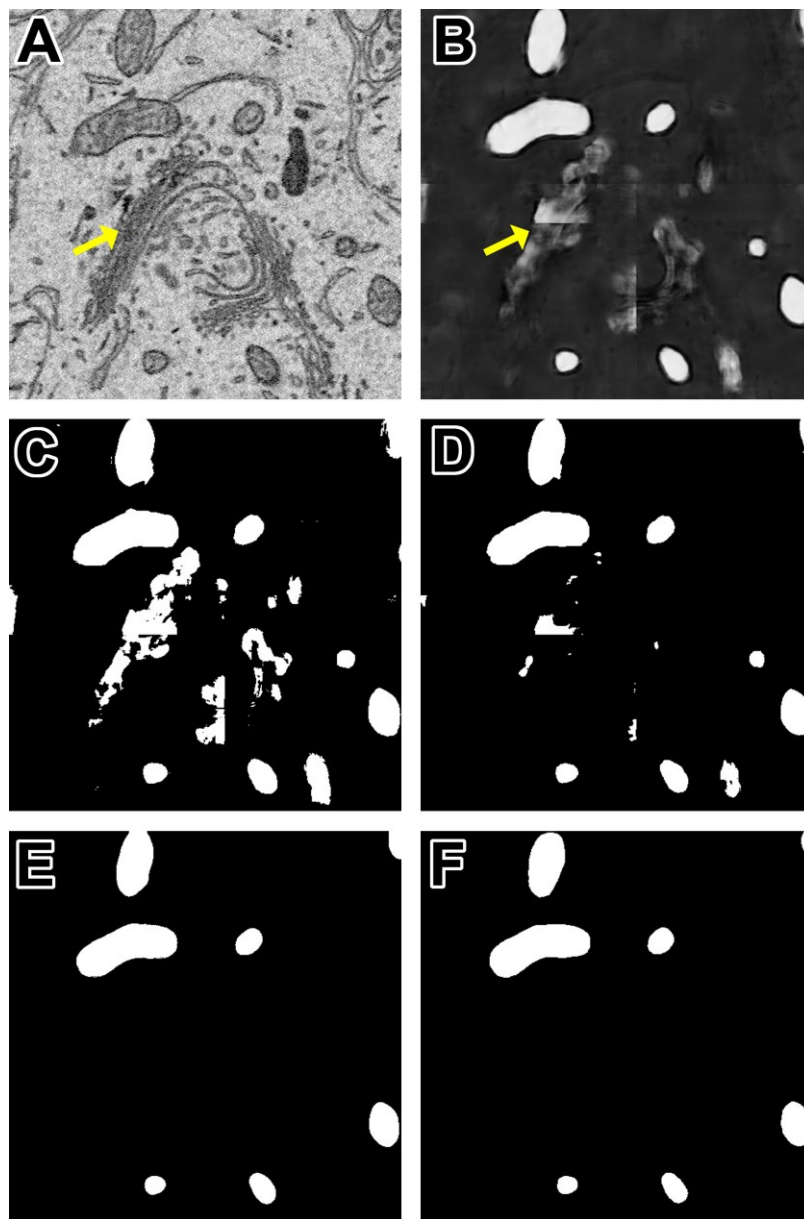
obtained for each stack using active contour segmentation were 0.985, 0.997, 0.972, and 0.979 for mitochondria, lysosomes, nuclei, and nucleoli, respectively.

A comparison of the proposed active contour binarization method to the other methods tested is shown in Figure 2.11 using mitochondria as an example. Since the Golgi apparatus can sometimes display a texture similar to that of the mitochondrial matrix, the presence of this organelle can confuse the mitochondrial classifier (Figures 2.11A and 2.11B, arrows). Segmentations generated with the maximum entropy algorithm (Figure 2.11C, recall = 0.992, precision = 0.498, F-value = 0.670, accuracy = 0.948) and Otsu's single-level method (Figure 2.11D, recall = 0.958, precision = 0.687, F-value = 0.812, accuracy = 0.977) retain elements of the Golgi apparatus as false positives. However, probability map binarization using the proposed active contour method eliminates these false positives (Figure 2.11D, recall = 0.908, precision = 0.804, F-value = 0.863, accuracy

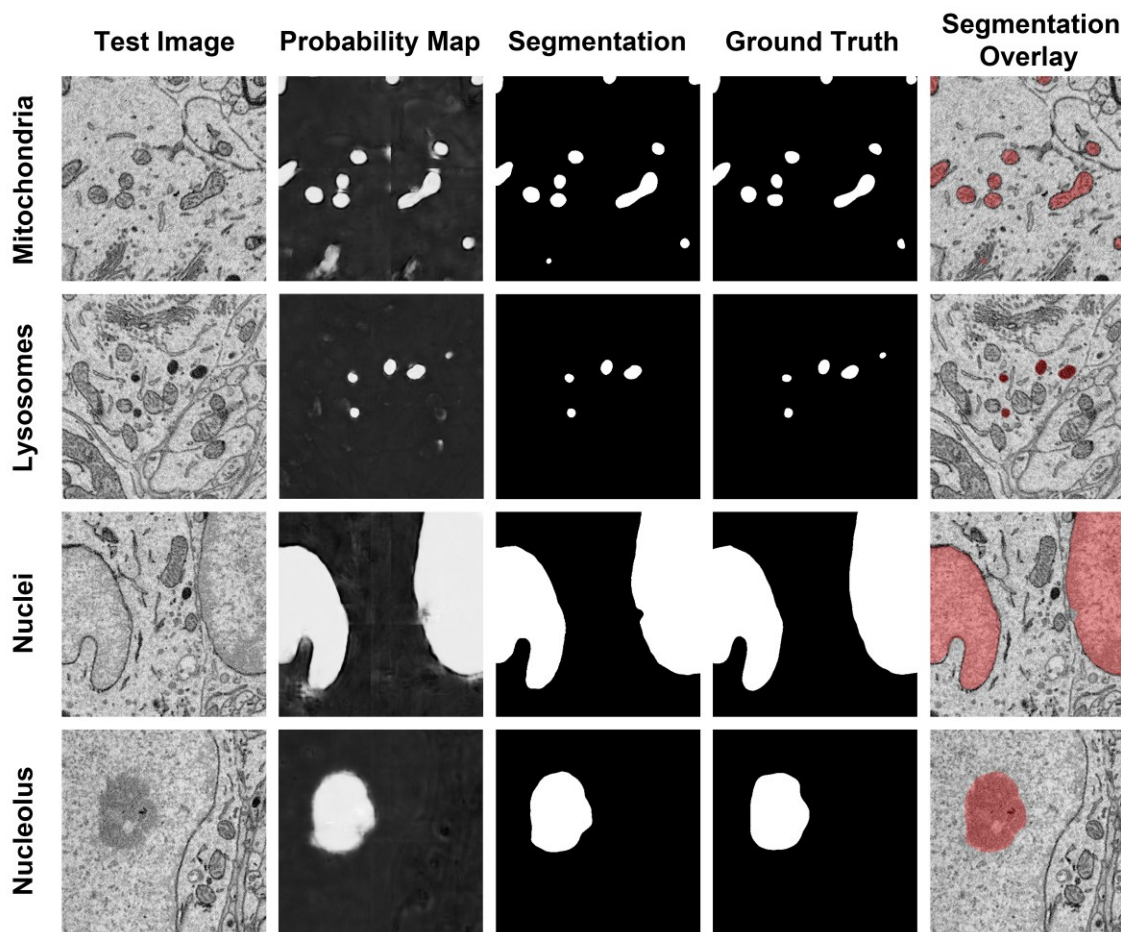
= 0.985) when compared to the ground truth (Figure 2.11E). Output probability maps and active contour segmentations from example test images of each organelle are shown in comparison to their corresponding ground truth in Figure 2.12.

#### **2.2.2.6. Comparison to a previously published algorithm**

The results of the proposed method for nuclear segmentation were validated by comparison to the results obtained by the algorithm of Tek and colleagues (Tek, et al., 2014). The full dataset was first downsampled to isotropic voxel dimensions (30 nm x 30 nm x 30 nm), resulting in a stack of size 4029 x 3120 x 1283 voxels. Training data and images consisted of a 500 x 500 x 50 subvolume of the downsampled stack containing two adjacent nuclei. Ground truth data were generated by manual segmentation of all neuronal, glial, and endothelial cell nuclei across fifty consecutive slices from the center of the dataset. A CHM pixel classifier with two stages and two levels was trained and applied to all images in the stack. Similarly, an ilastik voxel classifier was trained using all possible features with the same training images serving as input (Sommer, et al., 2014). This classifier was subsequently applied to all images in the downsampled stack. CHM probability maps (Figure 2.13) were binarized using the proposed method. The ilastik probability maps (Figure 2.14) were binarized by thresholding at the level  $p = 0.5$ , followed by the application of the object detection algorithm of Tek and colleagues with  $V_{th1}$  and  $V_{th2}$  set to 25 and 10000, respectively (Tek, et al., 2014). The proposed method achieved a precision, recall, and F-value of 0.976, 0.977, and 0.977, respectively. The method of Tek and colleagues achieved a precision, recall, and F-value of 0.976, 0.542, and 0.697, respectively, when applied to the same dataset using the same training data. ROC and precision-recall curves for the CHM and ilastik classifiers are given in Figures 2.15 and 2.16, respectively.

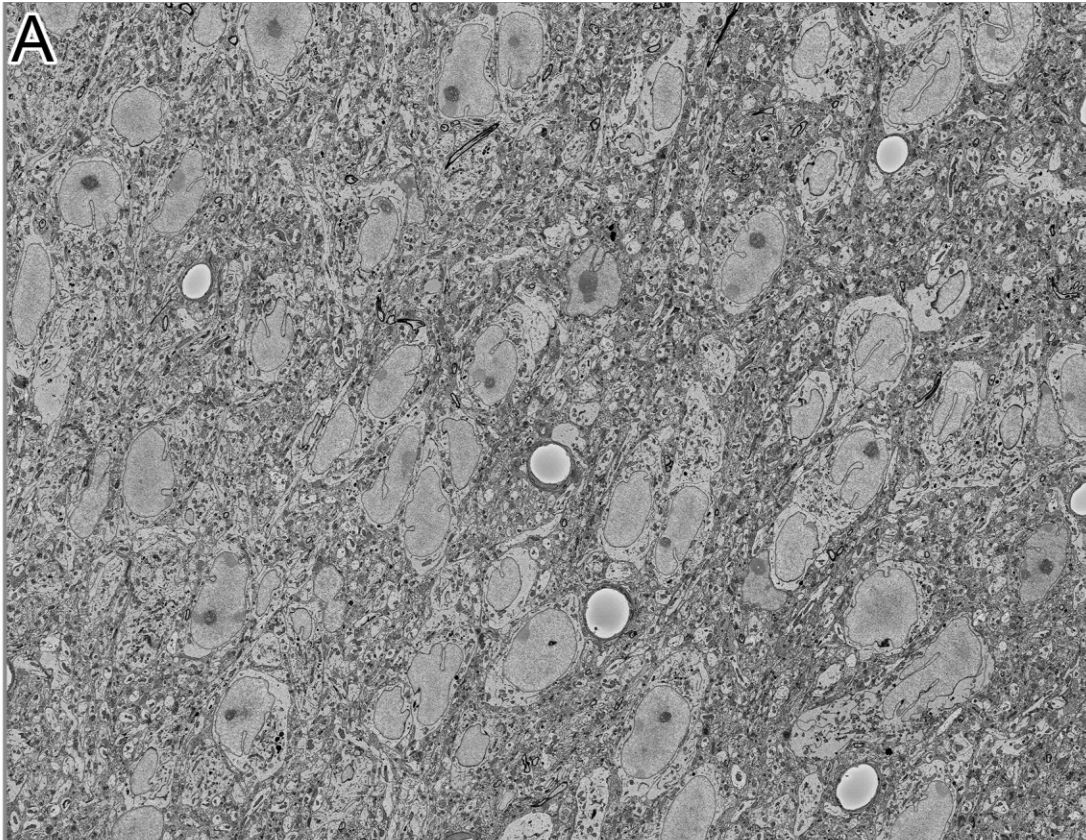


**Figure 2.11. Binarization of probability maps using active contours outperforms other methods.** A CHM classifier for mitochondria was applied to a 500 x 500 pixel test image (A), generating the probability map shown in (B). Note that regions of pixels corresponding to the Golgi apparatus (yellow arrows) were detected in the probability map. The Golgi apparatus can often confuse mitochondrial pixel classifiers because it has a texture very similar to that of the mitochondrial matrix. The results of binarization of the probability map using maximum entropy (C) and Otsu's single-level method (D) are shown. Using these techniques, regions of the Golgi are permitted into the final segmentation as false positives. The resultant segmentation obtained by our method of binarization with active contours ( $G = 2$ ,  $\alpha = 100$ ,  $\lambda = 8$ ) is shown in (E). Instances of the Golgi apparatus were automatically removed during processing. This segmentation (F-value = 0.863, accuracy = 0.985) is a highly faithful representation of the ground truth (F).



**Figure 2.12. The results of the proposed method are consistent when applied to diverse organelle targets.** The application of our method to different organelle targets yields consistent results without the need to significantly change the input parameters. Shown here are test images, each of size 500 x 500 pixels, and their corresponding probability maps, segmentations, and manually segmented ground truth images. The final column shows a transparent overlay of the segmentation onto the test image. The evaluation metrics for each test image are as follows: Mitochondria, F-value = 0.844, accuracy = 0.984; lysosomes, F-value = 0.872, accuracy = 0.997; nuclei, F-value = 0.971, accuracy = 0.971; nucleoli, F-value = 0.91, accuracy = 0.977.

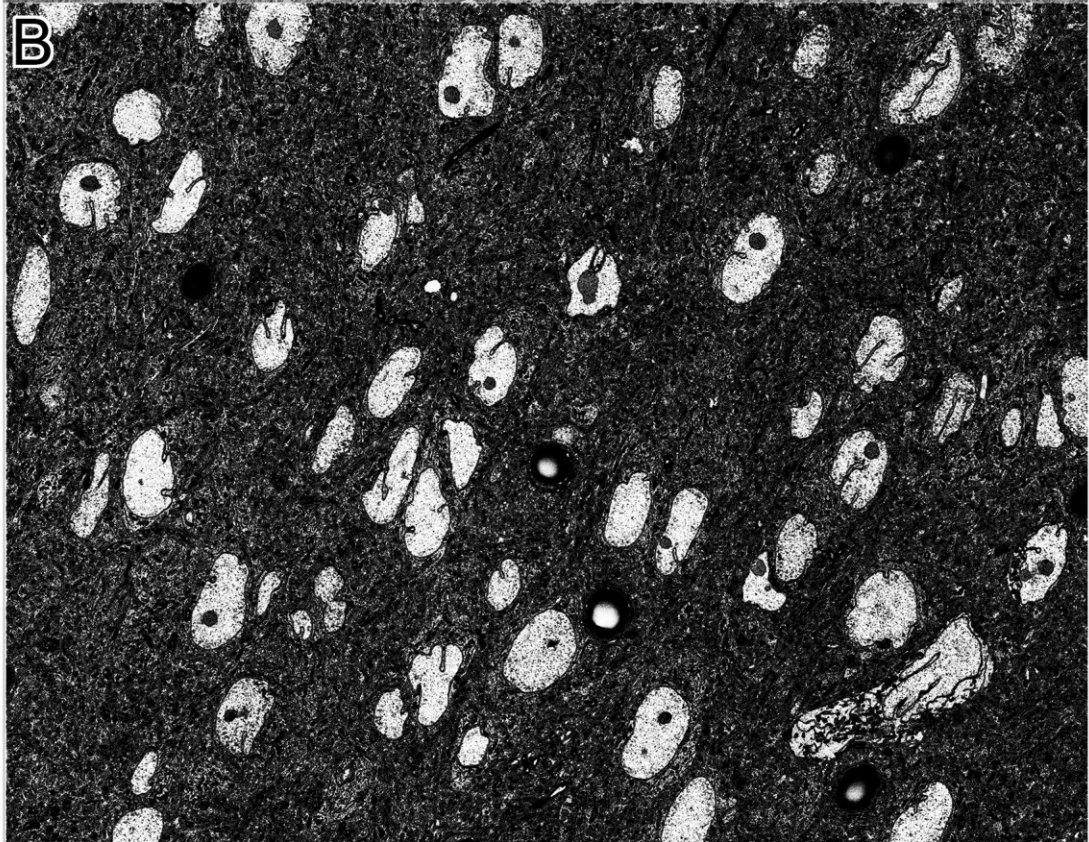
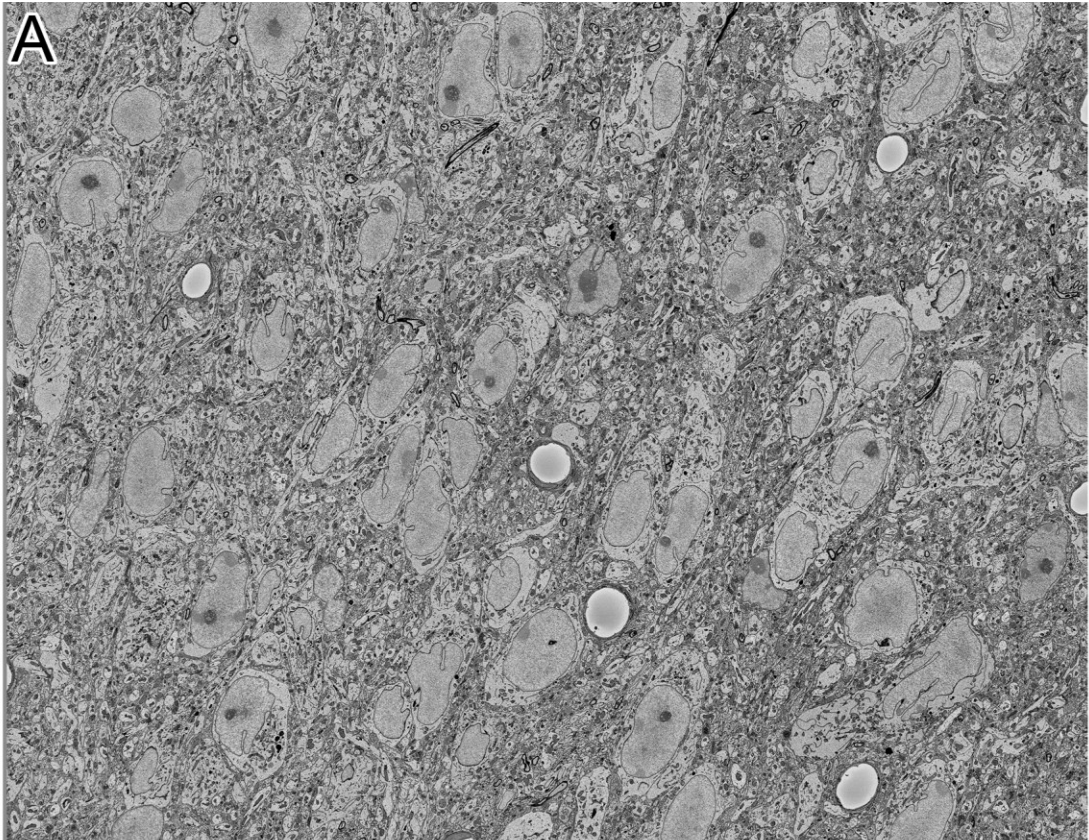
**Figure 2.13. A full-slice CHM probability map for nuclei.** The full dataset was first downsampled to isotropic voxel dimensions (30 nm x 30 nm x 30 nm), resulting in a stack of size 4029 x 3120 x 1283 voxels. Training data and images consisted of a 500 x 500 x 50 subvolume of the downsampled stack containing two adjacent nuclei. These training data were used to train a CHM pixel classifier with two stages and two levels, which was then used to generate full-slice probability maps for all 1,283 images in the stack. Shown here is a single slice from the stack (A) and its corresponding nuclear probability map (B).

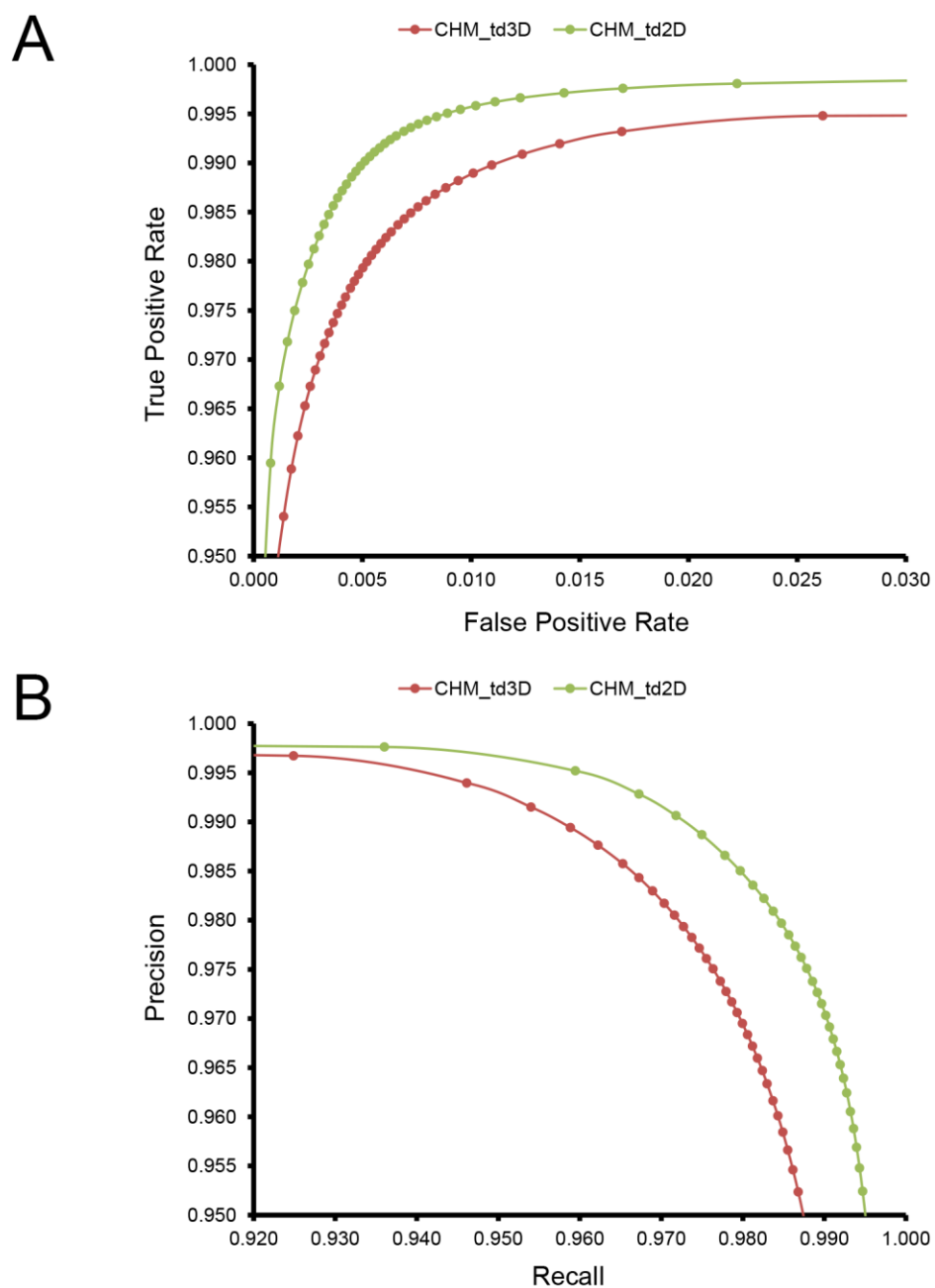




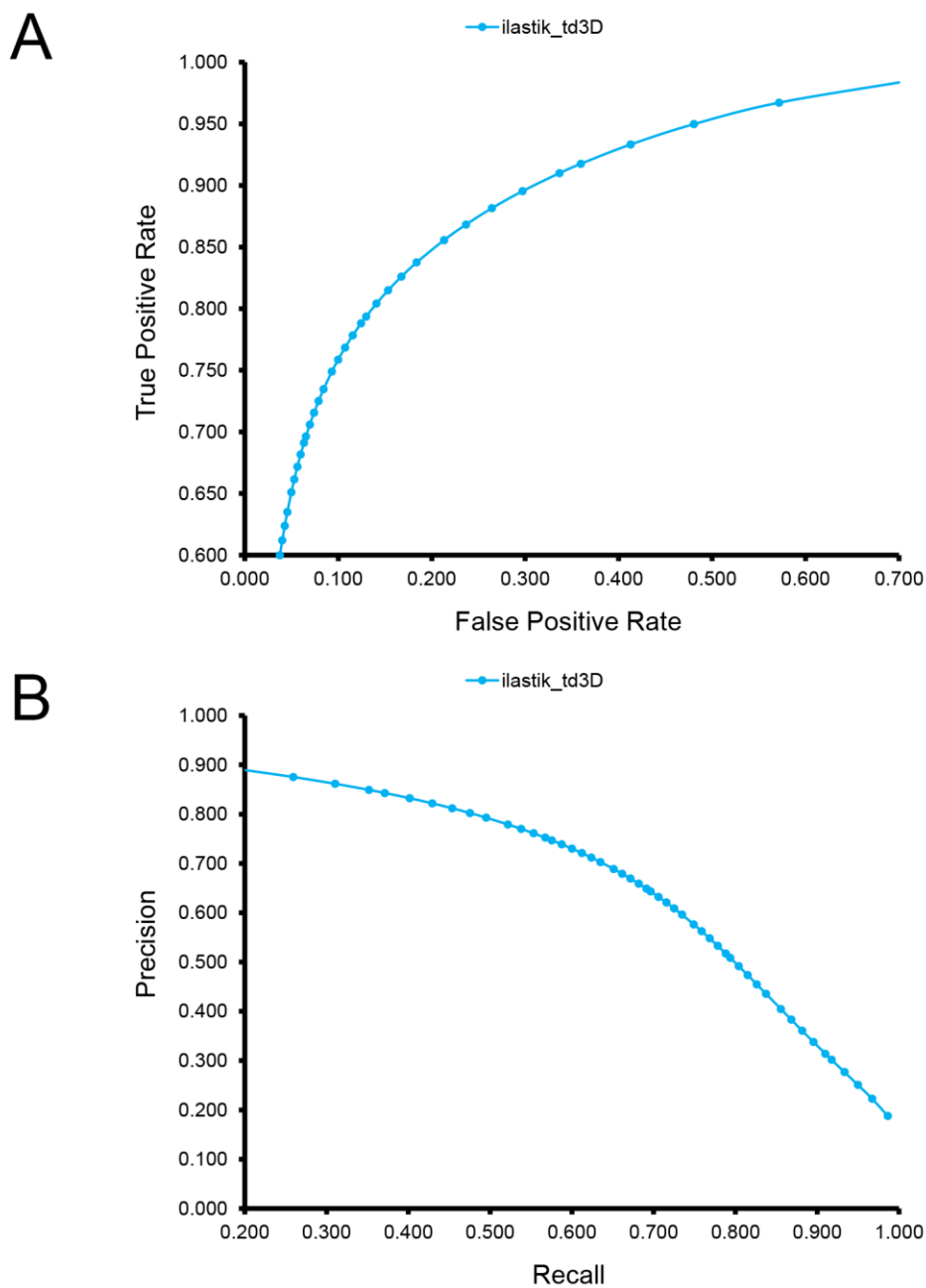
**Figure 2.14. A full-slice ilastik probability map for nuclei.** An ilastik voxel classifier was trained using the same training set used to produce the pixel classification show in Figure 2.13. All possible features and scales were included during classification. Shown here is a single slice from the stack (A) and its corresponding nuclear probability map (B). This probability map is significantly noisier than the one generated with the CHM.







**Figure 2.15. ROC and precision-recall curves for CHM nuclear classifiers.** The performance of a CHM classifier trained on a 3D subvolume of data (red) is compared to that of a CHM classifier trained on an equivalent amount of data collected from 2D tiles that were distributed throughout the volume (green). Both ROC (A) and precision-recall (B) curves demonstrate better performances for the classifier trained on 2D data. The performances of both classifiers were verified against the same ground truth.



**Figure 2.16. ROC and precision-recall curves for an ilastik voxel classifier.** The ilastik classifier evaluated here was trained using the same training data and evaluated against the same ground truth as the CHM\_3D classifier in Figure 2.15. The classifier was trained using the “headless” mode of operation. Its performance, as evaluated by ROC (A) and precision-recall (B) curves is significantly worse than that of both CHM pixel classifiers shown in Figure 2.15. It is possible that ilastik may perform better with less labels or a different proportion of label to background in its provided training labels.

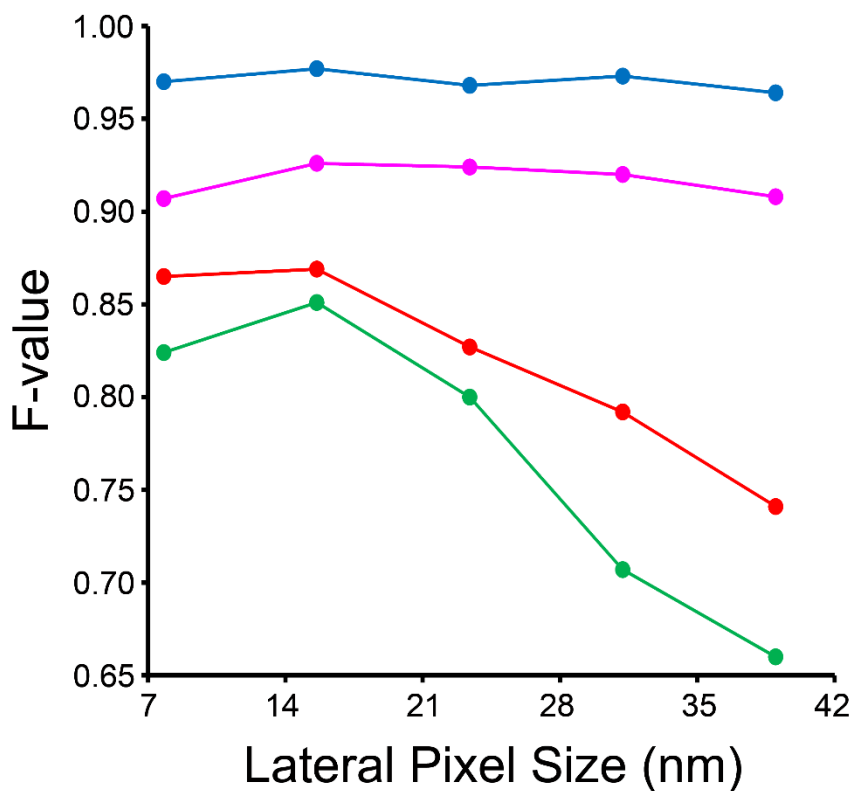
### **2.2.2.7. The impact of image downsampling on automatic segmentation performance**

The results of the downsampling experiment are shown in Figure 2.17. The resultant F-value for segmentation of nuclei and nucleoli remains remarkably consistent across the whole range of pixel sizes tested. The F-values for mitochondria and lysosomes exhibit substantial reductions at pixel sizes greater than  $\sim 15$  nm/pixel, corresponding to an overall downsampling of the original SBEM stack by a factor of four. The persistence of a high F-value across all scales tested for nuclei and nucleoli is likely due to their larger size and more regular texture in comparison to the other organelles. This is especially true for mitochondria, whose cristae architectures may differ dramatically from region to region.

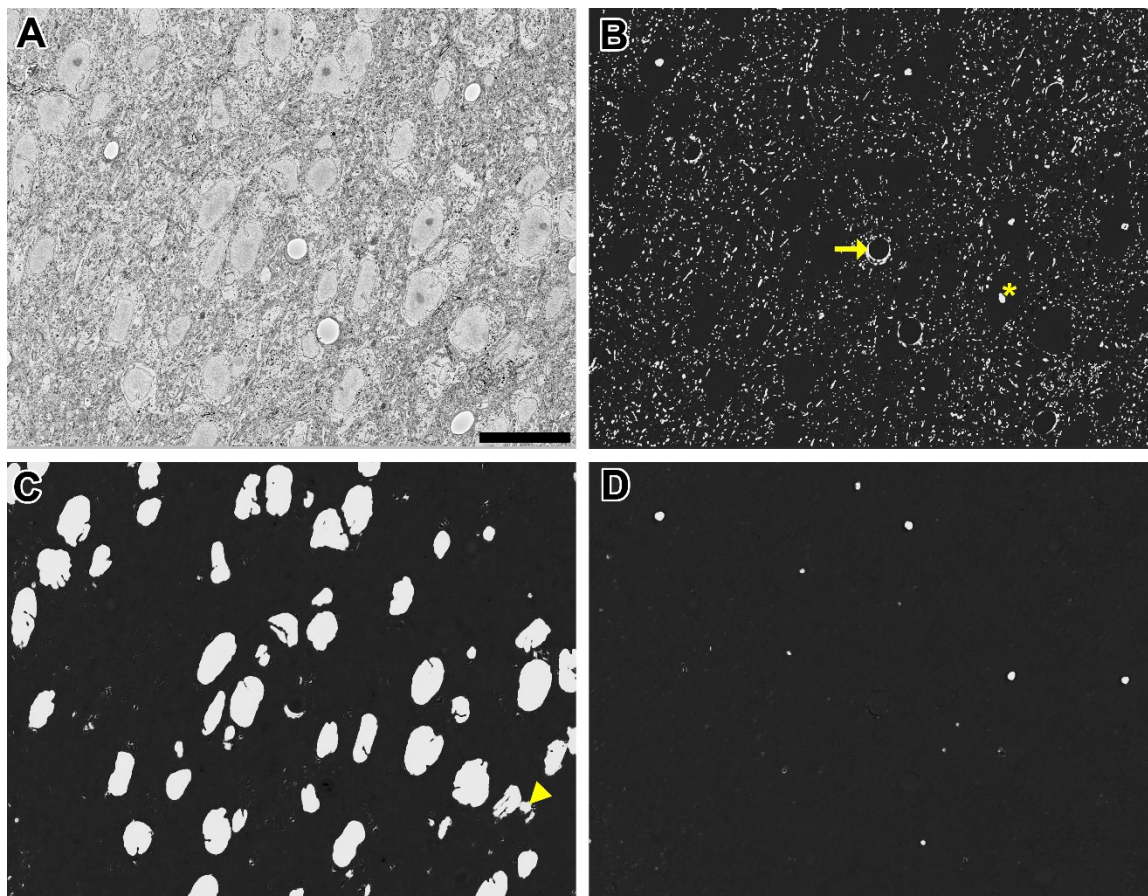
The required wall clock time and random access memory (RAM) required for CHM classifier training and pixel classification for each organelle at each level of downsampling were given in Table 2.2. The time and RAM required for probability map binarization are not shown because they are negligible with respect to training and classification. These results indicate that, in cases where segmentation accuracy is not dramatically affected, a vast amount of time and computational resources can be saved by downsampling the input image stacks. Simple extrapolation of pixel classification times shows that the time required by a single CPU to apply a nuclear pixel classifier to our full test dataset would be reduced from  $\sim 5.9$  years to  $\sim 0.4$  years when the input data are downsampled by a factor of 10. Examples of full image probability maps of nuclei, nucleoli, and mitochondria generated from downsampled data are shown in Figure 2.18.

## **2.3. Discussion**

As recently as a few years ago, the notion of reconstructing and morphologically characterizing the organelle networks of even a few whole cells was considered a



**Figure 2.17. Input images can be downsampled to various degrees before the segmentation results are negatively affected.** Each organelle-specific stack was downsampled by factors of two, four, six, eight, and ten. Separate classifiers were trained at each different pixel size and segmentations were generated for each stack using our method. Here, the F-value of each resultant stack is compared across the different pixel sizes obtained after downsampling. The F-value of nuclei (blue) and nucleoli (magenta) is remarkably independent of the level of downsampling across all levels tested. The F-values for mitochondria (red) and lysosomes (green) significantly decline as the level of downsampling is increased.



**Figure 2.18. Automatic segmentation can be efficiently scaled to handle full slices from teravoxel-sized SBEM datasets.** Probability maps of full images from the SCN dataset were generated by downsampling the image, computing probability maps of individual tiles, and stitching these tiled maps together. Shown here are probability maps of mitochondria (B), nuclei (C), and nucleoli (D) computed from the same full slice (A). The full slice was downsampled by a factor of two prior to mitochondrial pixel classification and a factor of eight before nuclear and nucleolar pixel classification. Common residual errors during mitochondrial pixel classification are the false detection of endothelial cells (arrow) and nucleoli or clusters of chromatin in the nucleus (asterisk). A common error encountered during nuclear pixel classification is the false detection of regions of cytoplasm devoid of membrane-bound organelles (arrowhead). These residuals are frequently removed by the application of the proposed probability map segmentation algorithm. Scale bar = 20  $\mu\text{m}$ .



monumental challenge (Noske, et al., 2008). The advent and widespread adoption of high throughput, volumetric EM techniques has threatened to change that notion, with the caveat that our ability to segment and analyze data must first catch up with our ability to collect it. With that goal in mind, this study aimed to develop a method for the accurate automatic segmentation of organelles in EM image stacks that: (1) could be easily adapted to any organelle of interest, and (2) could be applied to teravoxel-sized datasets in a computationally efficient manner.

Since it does not make any large-scale, *a priori* assumptions about the morphology of the segmentation target, the proposed method can be applied to segment diverse organelles with ease. The only geometrical properties assumed throughout the method are boundary smoothness and a cross-sectional area that is sufficient enough to prevent the removal of true positives following binary shrinking. Both of these assumptions are valid for virtually all organelles under practical imaging conditions. CHM classifiers can be trained for any dataset or organelle target if given the proper training data, and the output segmentations from the proposed method can be tuned to the demands of unique experiments. For example, decreasing the number of gray levels,  $G$ , used in the multi-level Otsu thresholding step will emphasize true positives at the expense of including false positives, which can often be excluded by post-processing filters. Additionally, it is easier to remove false positives by manual correction or crowd-sourcing (Giuly, et al., 2013) than it is to add missing true positives.

The proposed method performed favorably when compared to a recently published algorithm for the automatic segmentation of cell nuclei (Tek, et al., 2014). It is interesting to note that the performance of the proposed method was very similar when trained using either images from consecutive slices of the same nuclei (precision = 0.976, recall = 0.977) or single slice images from a variety of nuclei (precision = 0.973, recall = 0.968). This

similarity demonstrates the robustness of the CHM pixel classifier for this task. It is likely that the segmentation results obtained by applying the method of Tek and colleagues to the SCN dataset could be strengthened by training an ilastik voxel classifier against a greater diversity of nuclei.

Another advantage of the proposed method lies in its scalability to full datasets. The generation of probability maps from small tiles of the input image minimizes the required RAM. Additionally, it allows for computation to be easily expedited by parallelizing the processing of individual tiles across multiple CPUs. The demonstration that accurate results for certain organelles can be achieved on downsampled stacks also helps expedite processing. One can envision an experiment in which a teravoxel-sized SBEM stack collected at high resolution for axon tracking can then be downsampled and have its nuclei or mitochondria automatically segmented at a fraction of the computational cost that would have been required at its native resolution. As innovative methods to rapidly acquire even larger datasets continue to be developed (Helmstaedter, et al., 2013; Marx, 2013; Mohammadi-Gheidari and Kruit, 2011), this reduction in computational cost will prove critical.

In conclusion, the technologies proposed in this chapter introduce novel methods for the automatic segmentation of organelles from EM image stacks that are both robust and able to handle datasets of any size. These tools fill a critical need by allowing for the quantitative analysis of volumetric EM datasets at a scale between that of current connectomics approaches (Kim, et al., 2014; Helmstaedter, et al., 2013; Anderson, et al., 2011; Bock, et al., 2011; Briggman, et al., 2011; Kleinfeld, et al., 2011; Varshney, et al., 2011; Briggman and Denk, 2006) and that afforded by genetically encoded markers for small molecule localization (Boassa, et al., 2013; Martell, et al, 2012; Shu, et al., 2011).



This chapter, in part, is a reprint of the material as it appears in *Frontiers in Neuroanatomy*, 2014, 8. Perez, A.J., Seyedhosseini, M., Deerinck, T.J., Bushong, E.A., Panda, S., Tasdizen, T., and Ellisman, M.H. The dissertation author was the primary investigator and author of this paper.

### **Chapter 3**

## **From Pixels to Structures: Constructing Models of Neuronal Microanatomy**

### 3.1. Introduction

Understanding the relationship between structure and function at the subcellular level is of fundamental importance to biology. Organelle positioning within cells is thought to follow non-random organizational schemes that are rooted in the functionality of molecular-scale signaling cascades (de Brito and Scorrano, 2010). Consequently, it is not difficult to envision a number of biological questions that could be addressed with the aid of easy access to high resolution models and quantifications of organelle ultrastructure. For example, changes in mitochondrial fission and fusion events are known to correlate with a number of neurodegenerative conditions, including Parkinson's, Huntington's, and Alzheimer's diseases (Bossy-Wetzel et al., 2003; Knott, A.B. et al., 2008; Su et al., 2010). A neuroscientist studying one of these diseases might, therefore, wish to explore mitochondrial defects at the ultrastructural level across hundreds of cells from diseased tissue. In another example, a biologist may hope to obtain a high resolution depiction of how the knockout of certain motor proteins affects organelle localization throughout the cell (Tanaka et al., 1988). In addition, such whole-cell 3D models would undoubtedly benefit members of the computational modeling community, who might use such reconstructions to provide geometrical constraints for the modeling of  $\text{Ca}^{2+}$  diffusion or neurotransmitter release (Slepchenko et al., 2003; Buck et al., 2012).

As discussed in detail previously, the generation of the segmentations needed to produce such models requires a significant time investment in the form of either human or computational hours. Fortunately, the technologies described in Chapter 2 leverage upon machine learning approaches to yield such segmentations with minimal human interaction. The accurate, pixel-based segmentations output by these methods lay the groundwork for large-scale studies of cellular microanatomies. However, the modeling of representative biological morphologies from these data requires their expansion to the

third dimension, a step that is not trivial at larger scales. In this chapter, a series of computational methods for accelerating this process will be presented. An expeditious approach for the computation of 3D meshes from 2D binary segmentations will be outlined. Additionally, novel algorithms designed to enhance 2D segmentations by incorporating inter-slice contextual information will be described

However, even once accurate 3D models have been attained, the extraction of useful quantitative data from them remains no simple task. Quantitative analyses typically require the use of specialized software such as IMOD, Amira, or Imaparis, and formatting the data into a structure recognizable by such programs may necessitate numerous intermediate steps and file conversions. Additionally, it is incumbent upon the user to ensure accuracy and store the results in a reliable, shareable, and reproducible format. Therefore, it is clear that the automation of these steps, from initiation to data reporting, would greatly enhance the accessibility of large-scale, quantitative analyses to the general scientific community. A few open-source software packages have been generated with this goal in mind (McComb et al., 2009). The MTK program contained in the IMOD distribution has been used for a variety of quantitative analyses, including the study of ribosome densities (Kang and Staehelin, 2008) and the distributions of microtubules (Austin et al., 2005), mitochondria (Höög et al., 2007), and synaptic vesicles (Gibeaux et al., 2013). Applications have been developed to automatically quantify and report label density in confocal datasets (Dayal and Hill, 2014) and morphological parameters from nuclei at the LM level (Ollion et al., 2013; Poulet et al., 2014). A recent contribution from Graham Knott and Pascual Fua, NeuroMorph, is an attempt at bringing such automated quantifications to the level of large-scale EM (Jorstad et al., 2014). Though this tool has been used for studying dendritic spines and synapses, there are no known analogous

tools for automating the analysis of organelle morphologies in teravoxel-sized 3D EM datasets.

With this need in mind, a final contribution of this chapter will be the description of a workflow for the automatic calculation and reporting of single-cell organelle morphologies and spatial distributions using the nucleus as a test case. Importantly, it will be demonstrated that these quantification steps can be linked to the automatic segmentation algorithms of Chapter 2 in a seamless workflow that automatically outputs numerical data following segmentation. Taken as a whole, this workflow represents a powerful tool that enables the quantification and modeling of subcellular microenvironments with high degrees of resolution and ease.

### **3.1.1. Nuclear structure and function**

The nucleus is generally the largest organelle found in eukaryotic cells and tends to be the most obvious and defining cellular feature when viewed at the microscopic level. According to both structural and functional criteria, the nucleus can be divided into two distinct compartments: (1) the nuclear envelope (NE) and (2) the nuclear interior, or nucleoplasm. The NE consists of two phospholipid bilayers, the inner (INM) and outer (ONM) nuclear membranes, which are separated by a roughly 30-50 nm wide lumen known as the perinuclear space (Zwerger et al., 2011). Large proteinaceous assemblies known as nuclear pore complexes (NPCs) are embedded across the nuclear surface. These complexes span both the INM and ONM, forming the sole gateway for the exchange of ions and macromolecules between the cytoplasm and nucleoplasm (Hoelz et al., 2011). At a mechanical level, the nucleus is physically coupled to the cell-wide cytoskeletal network via protein complexes that cross the perinuclear space and link cytoskeletal elements with the nuclear lamina, a meshwork of proteins that lines the nucleoplasmic

face of the INM (Crisp et al., 2006; Tzur et al., 2006; Wang et al., 2009). Importantly, the ONM and perinuclear space are continuous with the lumen of the ER, and this system provides a reservoir of membrane that can, in theory, be used to accommodate nuclear shape changes and deformations (Lammerding et al., 2007).

The main function of the second nuclear compartment, the nucleoplasm, is to sequester the cell's genetic material in the form of chromatin. The two distinct configurations of chromatin, heterochromatin and euchromatin, have been shown to occupy distinguishable and non-random regions of the nucleoplasm. Euchromatin, the more transcriptionally active of the two forms, tends to be found towards the nuclear interior whereas the less active heterochromatin is found closer to the nuclear periphery (Zwerger et al., 2011). Furthermore, it has been shown that individual chromosomes of interphase nuclei occupy defined regions of the nucleoplasm, called chromatin territories (Cremer and Cremer, 2001). The existence of a nuclear skeletal network that facilitates this compartmentalization has been proposed (Nelson et al., 1986; Ingber, 1997), but its existence is still the subject of much scientific debate.

In addition to chromatin, the nucleoplasm contains a number of other distinct structures that are readily identifiable at the EM level. The largest and most obvious of these is the nucleolus, which serves as the site of ribosomal RNA processing and ribosome biogenesis (Hetman and Pietrzak, 2012). The area surrounding the nucleolus, known as the perinucleolar compartment (Huang et al., 1997), is an ordered domain containing small RNAs and perinucleolar chromatin (Nemeth and Langst, 2011; Padeken and Heun, 2014). Coiled, or Cajal, bodies are another type of nuclear organelle, and these bodies are associated with states of cellular stress and are frequently localized in the perinucleolar vicinity (Cioce and Lamond, 2005). Nuclear speckles, named after their appearance when fluorescently labeled, contain large concentrations of small nuclear

ribonucleic particles (snRNPs; Handwerger and Gall, 2006). The location and composition of these organelles within the nucleoplasm changes in response to levels of mRNA transcription and protein phosphorylation (Lamond and Spector, 2003). The importance of these nuclear organelles, in both normal and diseased states, remains a topic of great interest within the community.

Disruptions of wildtype (WT) nuclear structure and mechanics are associated with a growing number of disease states. Changes in nuclear size and shape, as well as alterations in chromatin texture and nucleolar number, have long been used as markers for detecting tumor cells in clinical cancer diagnosis (Zink et al., 2004). The mutation of a single codon within the gene encoding the ER/NE resident protein torsinA leads to the development of the neurological movement disorder DYT1 dystonia (Tanabe et al., 2009). At the microscale, this mutation manifests as the selective blebbing of the ONM in neurons of mice, producing a grossly perturbed nuclear structure (Goodchild et al., 2005; Kim et al., 2010). Mutations in proteins of the nuclear lamina are associated with a set of human diseases collectively referred to as the laminopathies (Worman, 2012). The laminopathies typically result from mutations to the gene encoding the protein lamin A, and include Emery-Dreifuss muscular dystrophy, Hutchinson-Gilford progeria syndrome, and dilated cardiomyopathy (Capell and Collins, 2006). One hypothesis for the mechanism behind these diseases is that laminar mutations reduce the structural integrity of the nucleus, ultimately weakening it and leading to cell death in mechanically stressed tissues such as muscle (Zwerger et al., 2011).

One interesting structural characteristic of many interphase nuclei is the presence of deep grooves, or invaginations, of the nuclear surface. Such nuclear invaginations have been reported in a host of species and cell types, including plants (Li and Dickinson, 1986; Collings et al., 2000), yeast (Vitols et al., 1961), cultured 3T3 cells (Clubb and Locke,

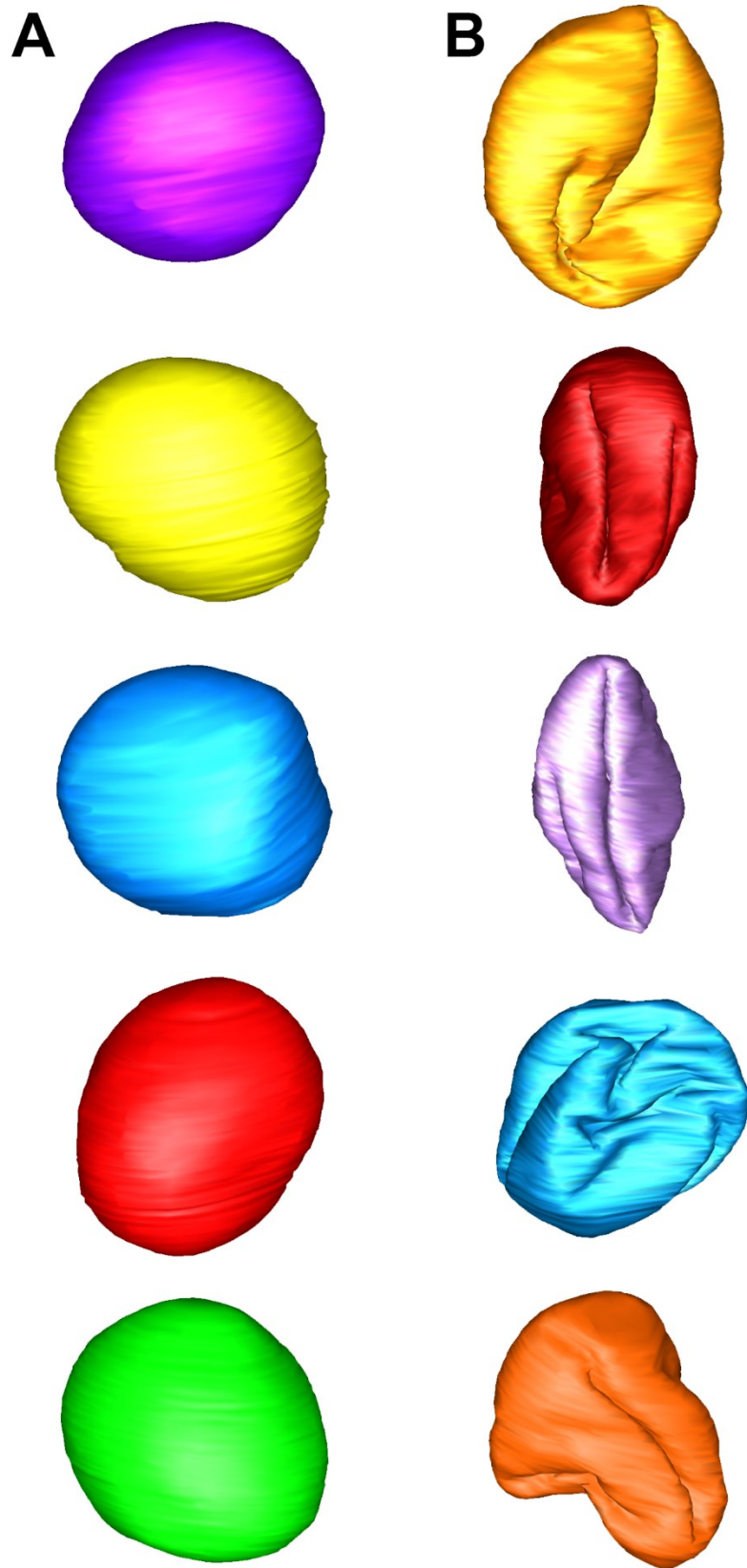
1998) and myocytes (Abe et al., 2004), and mammalian B cells (Dardick et al., 1982) and hypothalamic neurons (van den Pol, 1980). Nuclear invaginations can be separated into two categories based on their ultrastructural nature: Type I invaginations involve only extensions of the INM into the nucleoplasm, while Type II invaginations involve the similar extension of both nuclear membranes (Malhas and Vaux, 2014). Studies using high molecular weight fluorescent tracers confirmed that the lumens of invaginations are contiguous with the cytoplasm (Malhas et al., 2011), and the presence of NPCs in invaginated regions of the NE was confirmed by ssTEM (Fricker et al., 2007).

The widespread distribution of nuclei displaying the invaginated phenotype has led many to believe that these folds serve a critical purpose in cell biology. Nucleoli are observed to be preferentially associated with invaginations (Fricker et al., 2007), and the cytoplasmic lumens of invaginations often contain membrane-bound organelles and markers of translational initiation (Paytubi et al., 2009). Therefore, invaginations may serve to expedite the translation of critical mRNAs that are generated adjacent to the fold and exported to the cytoplasm by invagination-localized NPCs. (Malhas et al., 2011). In a recent report, Mauger suggested that nuclear invaginations may also affect cellular  $Ca^{2+}$  dynamics and serve to regulate transcriptional activity (Mauger, 2012). Whatever the purpose of invaginations may be, it is likely that they are advantageous in a cell-specific manner rather than necessary for normal cell function; populations of both invaginated and non-invaginated nuclei can be found in different regions of the healthy mouse brain (Figure 3.1).

Despite such interest in the nature of nuclear folding, there are currently few reliable methods for quantifying it in 3D. On account of this, some studies have reported purely qualitative observations, commenting on the percentage of a certain cell type that displays an invaginated phenotype (Abe et al., 2004). The most obvious descriptor for the



**Figure 3.1. Nuclear invaginations are specific to certain populations of cells.** Depicted here are nuclear surface renderings from neurons of the mouse CA1 hippocampus (A, column) and the mouse hypothalamic suprachiasmatic nucleus (B, column). Despite being reconstructed from the brain of the same species, nuclei from the CA1 hippocampus have almost perfectly smooth surfaces, while the nuclei from the suprachiasmatic nucleus are heavily invaginated. Therefore, by an unknown mechanism, the nuclear invagination phenotype is specific to certain populations of cell types. The surface renderings depicted here were automatically generated using the multiplane automatic segmentation algorithm described in this chapter.



degree of nuclear folding is the surface area to volume ratio (SVR), which should, in theory, be greater for more heavily invaginated nuclei (O'Connell et al., 2008). Lafarga and colleagues proposed a dimensionless invagination factor ( $IF_{2D}$ ) to quantify the extent of cross-sectional nuclear folding in ultrathin section TEM images. Their equation, in which  $A$  and  $P$  represent the cross-sectional area and perimeter of the nucleus, respectively, is given below (Lafarga et al., 1992):

$$IF_{2D} = \frac{P}{\sqrt{A}} \times K_{2D}$$

$$K_{2D} = \frac{1}{2\sqrt{\pi}}$$

The IF is normalized such that it furnishes a value of one for a perfect circle and values greater than one in the presence of any degree of folding. Numerous other 2D shape descriptors, such as circularity, solidity, and eccentricity may also yield useful correlations to the degree of nuclear folding (Choi et al., 2011). In this chapter, a number of shape descriptors for the accurate quantification of 3D nuclear morphologies will be introduced and applied to automatically generated surface renderings. But first, in the following sections, two methods for enhancing segmentation results will be introduced and their application to improving the accuracy of automatically generated nuclear morphologies will be emphasized.

## **3.2. Methods Development and Results**

### **3.2.1. The multiplane automatic segmentation algorithm**

Large structures with heterogeneous textures may be difficult to automatically segment using a single classifier. An example of such a structure is the nucleus, which contains unpredictably distributed clumps of chromatin with different textures and patch sizes. This can lead to the erroneous assignment of low probabilities to pixels at the

nuclear border, where clumps of heterochromatin accumulate. Such low probabilities result in false negatives in the output segmentation, which typically manifest as an uneven or jagged nuclear border (Figure 3.3). This error is harmful when a quantification of exact nuclear morphology is desired, as it will lead to an often substantial overestimate of nuclear surface area.

When an SBEM dataset is re-sliced about an orthogonal plane (Figure 3.2) and this re-sliced representation is used as input to the pixel classification process, the classifier is presented with a different view of the data that may yield better segmentation accuracy. However, it is impossible to know which orientation will provide the best results *a priori*, and it is likely an aggregate of many views that will achieve optimal classification. In this section, a method for improving the results of 2D automatic segmentation is presented. The input SBEM dataset is first downsampled to isotropic voxels, then re-sliced to yield views in the XZ- and YZ-planes. A single pixel classifier is applied to all three stacks, and the results are averaged together to give a final probability map stack. In the following sub-sections, the principles behind this method, named the multiplane automatic segmentation (MPAS) algorithm, will be outlined. This will be followed by a description of its implementation and some results that demonstrate its use to achieve improvements in output nuclear segmentations.

### 3.2.1.1. Description of the algorithm

Consider an 8-bit SBEM stack,  $I_{XY}$ , in its native orientation, with isotropic voxels and dimensions specified by  $(X, Y, Z) = (s_X, s_Y, s_Z)$ . Since SBEM stacks have a coarser resolution in their axial dimension, the isotropic voxel size of  $I_{XY}$  is equivalent to the axial step size,  $\delta$ . Since the stack  $I_{XY}$  is equivalent to a 3D matrix of pixel intensity values, it can be rotated by  $90^\circ$  about both its X and Y axes, yielding the rotated image stacks:

$$I_{XZ}(i, j, k) = \{0, \dots, 255\} \forall i \in \{1, \dots, s_X\}, j \in \{1, \dots, s_Z\}, k \in \{1, \dots, s_Y\}$$

$$I_{YZ}(i, j, k) = \{0, \dots, 255\} \forall i \in \{1, \dots, s_Y\}, j \in \{1, \dots, s_Z\}, k \in \{1, \dots, s_X\}$$

These rotated stacks depict views that are equivalent to slicing  $I_{XY}$  in increments of  $\delta$  nm about its XZ and YZ planes. Since voxels are isotropic, the texture, color, edge, and other pertinent image features of  $I_{XZ}$  and  $I_{YZ}$  are comparable in scale to those of  $I_{XY}$ . Therefore, only one trained model is required to reliably classify the pixels for all three stacks. The application of a trained CHM pixel classifier to each stack yields the probability map stacks  $P_{XY}$ ,  $P_{XZ}$ , and  $P_{YZ}$ :

$$P_{XY}(i, j, k) = \{0, \dots, 1\} \forall i \in \{1, \dots, s_X\}, j \in \{1, \dots, s_Y\}, k \in \{1, \dots, s_Z\}$$

$$P_{XZ}(i, j, k) = \{0, \dots, 1\} \forall i \in \{1, \dots, s_X\}, j \in \{1, \dots, s_Z\}, k \in \{1, \dots, s_Y\}$$

$$P_{YZ}(i, j, k) = \{0, \dots, 1\} \forall i \in \{1, \dots, s_Y\}, j \in \{1, \dots, s_Z\}, k \in \{1, \dots, s_X\}$$

To combine results, the stacks  $P_{XZ}$  and  $P_{YZ}$  must first be converted back to the native, XY orientation. This is achieved by applying the reverse 90° rotations about the X and Y axes, giving the rotated stacks  $P'_{XZ}$  and  $P'_{YZ}$ , respectively. The average probability map stack is attained by taking the voxel-by-voxel geometric mean (gmean) of all three probability map stacks in the XY orientation:

$$\bar{P}(i, j, k) = \text{gmean}(P_{XY}(i, j, k), P'_{XZ}(i, j, k), P'_{YZ}(i, j, k)) \forall i \in \{1, \dots, s_X\}, j \in \{1, \dots, s_Y\}, k \in \{1, \dots, s_Z\}$$

This average probability map is then binarized using the method described in Section 2.2.2.5, giving the final segmentation:

$$S(i, j, k) = \{0, \dots, 1\} \forall i \in \{1, \dots, s_X\}, j \in \{1, \dots, s_Y\}, k \in \{1, \dots, s_Z\}$$

### 3.2.1.2. Implementation and Results

The SBEM stack in its native XY orientation is first downsampled to isotropic voxels using the method described in Chapter 2.2.1.3. Re-sliced representations depicting views

in the XZ and YZ planes are generated using the IMOD program *rotatevol*. Each of these MRC stacks is then converted to a sequence of PNG images using the method described in Chapter 2.2.1.3. The PNG stacks from all three representations are subsequently classified using a CHM pixel classifier according to the methods described in Chapter 2.2.2. MPAS post-processing is initiated using the script *mpas.sh* (Appendix C.3.1). This script requires paths to the three directories containing the full-dataset probability maps generated previously (XY, XZ, and YZ). In the first step, all PNG images are converted to single-slice MRC files using the script *mpas\_png2mrc.q* (Appendix C.3.2). Single-file MRC stacks are made from each set of single-slice MRCs, and the XZ and YZ MRC stacks are rotated back into the XY orientation using the script *mpas\_stackandRotate.q* (Appendix C.3.3). Once rotation has completed, all three stacks are averaged together with the script *mpas\_average.q* (Appendix C.3.4). All of these steps are performed automatically and in parallel once the *mpas.sh* script has been initiated.

Examples of XY, XZ, and YZ probability maps generated using the MPAS process are shown in Figure 3.2. Consistent pixel classifications were achieved for all orientations using a classifier trained on images from the XY orientation only. The use of only one classifier is advantageous because it reduces the computational time needed for training as well as reduces the manual segmentation time necessary for generating training data about all orientations. A colored representation of three orthogonal probability maps from the same nucleus (Figure 3.3A) further demonstrates this consistency. Figure 3.3B illustrates the improvements afforded by the MPAS algorithm in the context of nuclear segmentation. Averaging from multiple probability maps using the MPAS algorithm results in a better segmentation by filling in pixels at the nuclear periphery that were assigned low probabilities by the single XY classifier alone. Table 3.1 gives a quantitative example of this improvement in the context of parameters of nuclear morphology. The volume and

**Table 3.1. A nuclear segmentation generated automatically by MPAS yields a more faithful representation of ground truth morphology.** The volume and surface area output automatically by single orientation segmentation and MPAS are compared to those of manually segmented ground truth. The nuclear rendering generated automatically by MPAS has a volume and surface area much closer to those of the ground truth. The biggest gain is seen in the quantification of surface area, for which MPAS is significantly more accurate.

	Volume ( $\mu\text{m}^3$ )	dV ( $\mu\text{m}^3$ )	dV (%)	Surface Area ( $\mu\text{m}^2$ )	dS ( $\mu\text{m}^2$ )	dS (%)
Ground Truth	317.39	-	-	368.4	-	-
XY only	312.79	-4.60	-1.45	401.17	+32.71	+8.90
MPAS	315.37	-2.02	-0.64	376.95	8.65	+2.35

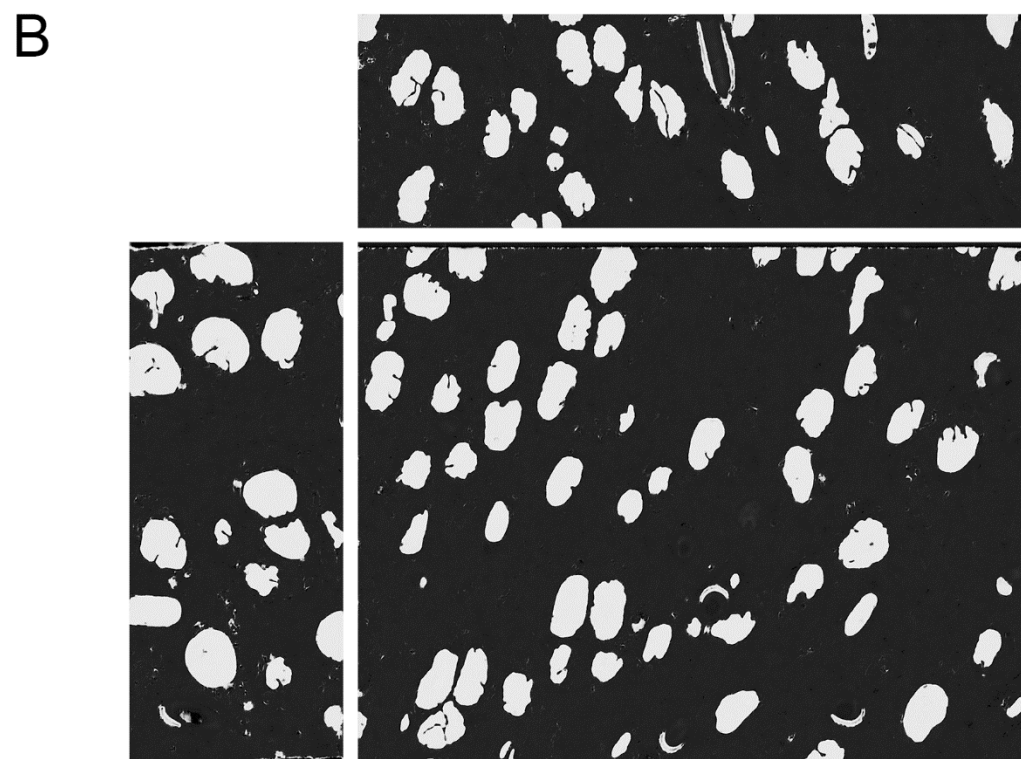
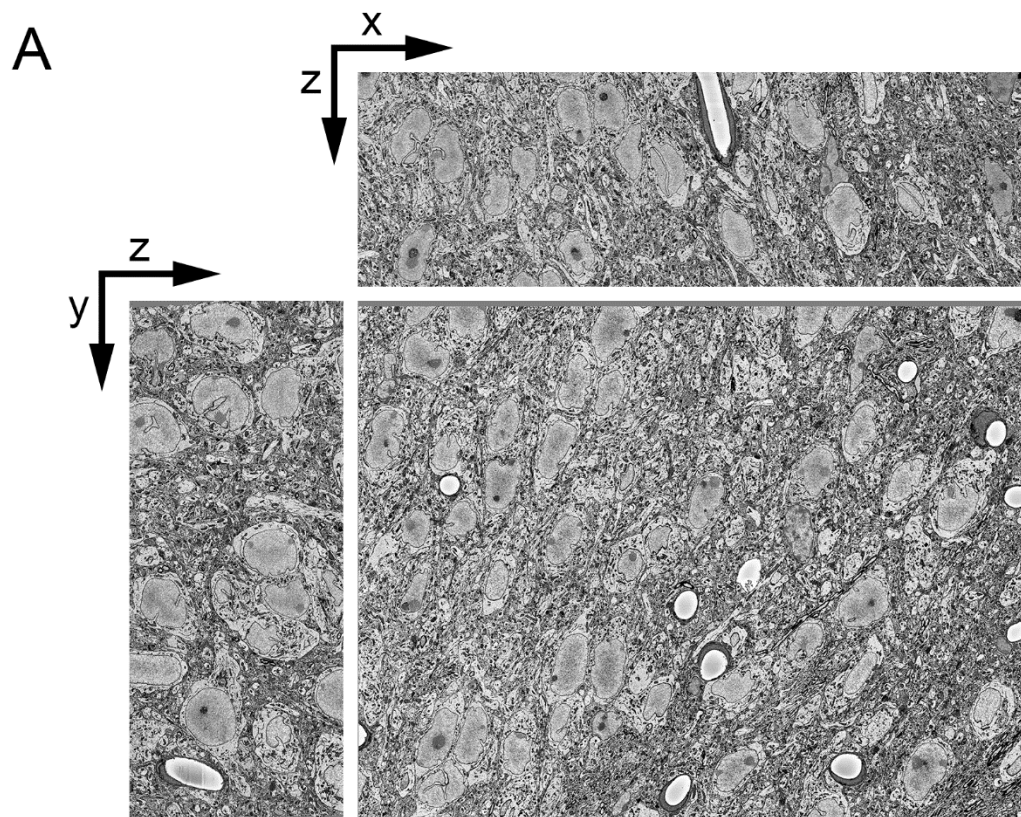
surface area of a nucleus were compared for renderings generated from (1) a manual segmentation, (2) a XY CHM segmentation, and (3) a MPAS CHM segmentation. The volume and surface area of the manually segmented surface were considered to be ground truth. The surface rendering generated automatically from the MPAS process had a volume and surface area much closer to those of the ground truth, yielding a far more accurate representation of true nuclear morphology than the single orientation segmentation. As anticipated, the largest gain of using the MPAS algorithm was made in the quantification of nuclear surface area. The single orientation automatic segmentation furnished a surface area that was 8.9% larger than ground truth, while the surface area generated by MPAS was only 2.35% greater.

### 3.2.2. Interslice interpolation of 3D objects

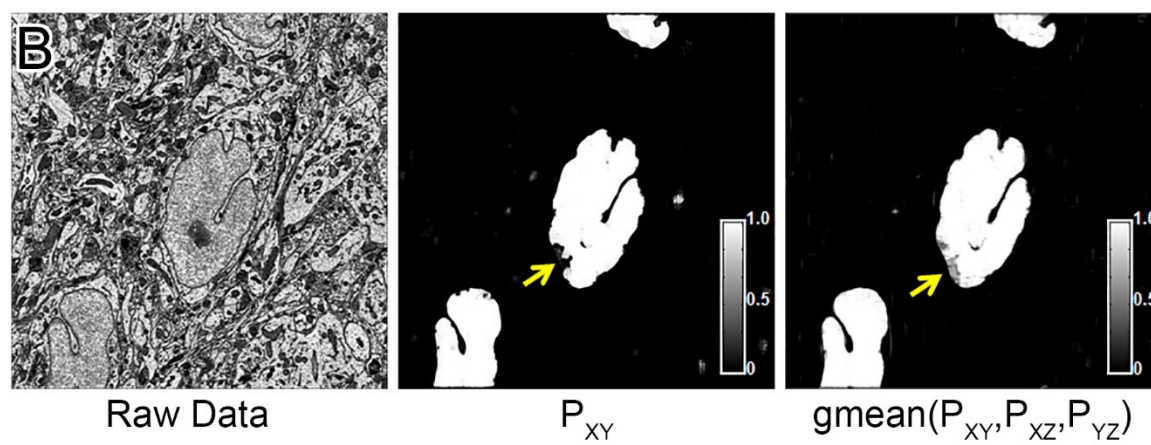
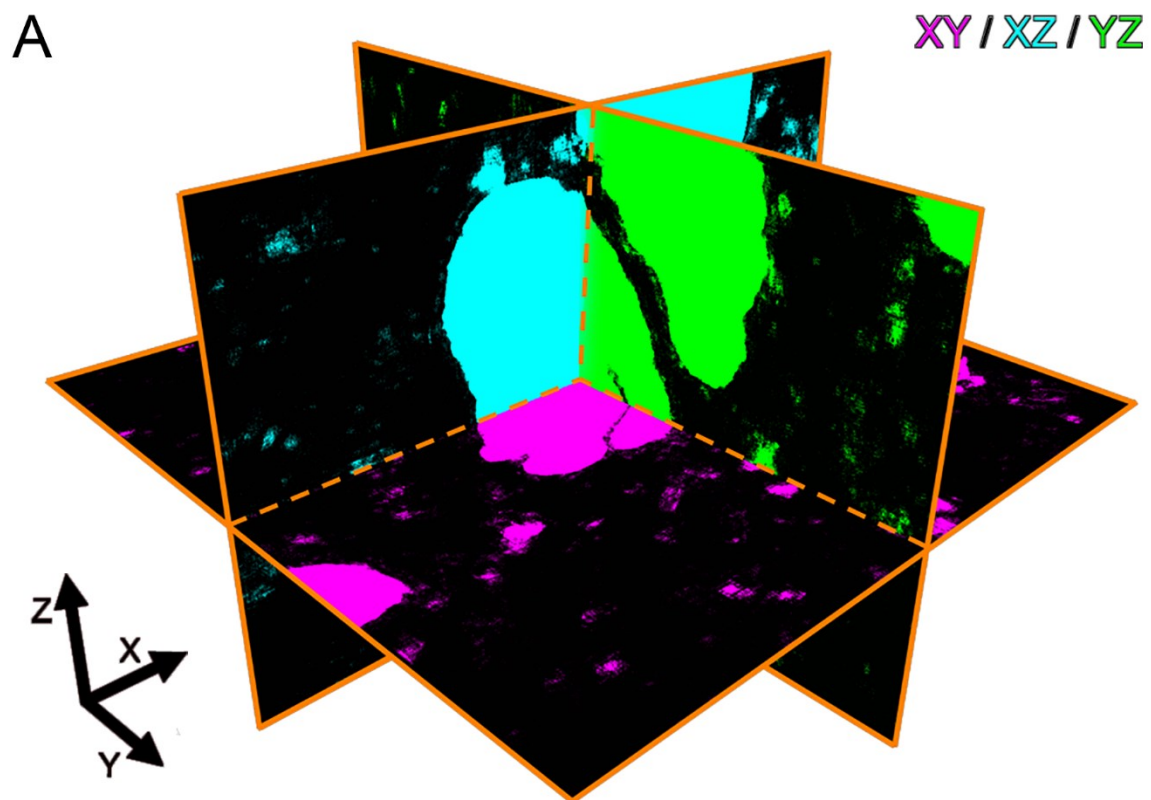
One of the most significant drawbacks of automatic segmentation methods is that, even in the most ideal cases, some degree of manual correction is needed when the accurate quantification of single organelle morphologies is desired. One way to reduce, or potentially eliminate, this need is to implement an automatic method to detect poorly segmented slices and replace them with interpolations between slices that are presumed

**Figure 3.2. The MPAS algorithm produces accurate probability maps for re-sliced data in different orientations using a single pixel classifier.** Shown here are the raw XY, XZ, and YZ images (A) from an isotropic SBEM dataset. Since isotropic voxels were used, the XZ and YZ orientations do not appear to be compressed and demonstrate quality and features similar to those of the XY orientation. Shown below (B) are the CHM nuclear probability maps generated for each of the slices shown in (A). All probability maps were generated using the same classifier, which was trained on data solely in the XY orientation.





**Figure 3.3. The application of MPAS to the automatic segmentation of nuclei helps properly classify pixels near the nuclear envelope.** MPAS was used to segment nuclei from the SCN dataset. Shown in (A) are three orthogonal slices through a nucleus, demonstrating probabilities obtained by CHM pixel classification about the XY, XZ, and YZ planes. MPAS automatic segmentation helps to fill in higher probabilities to the boundaries of nuclei that may have been improperly classified by considering only one orientation (B, yellow arrows). Such misclassifications are typically the result of patches of heterochromatin and other inconsistent features of nuclei. MPAS helps alleviate these misclassifications by incorporating averages over multiple views of the same object.



to be properly segmented. In this section, an algorithm for generating accurate interslice interpolations between input and output 2D binary images will be described and tested. It will then be applied to correct segmentations of nuclei generated using the methods of the previous chapter.

The proposed method is inspired by the morphological skeleton interpolation (MSI) algorithm of Chatzis and Pitas (Chatzis and Pitas, 2000). The MSI algorithm consists of five principal steps: (1) Object skeletonization, (2) skeleton matching, (3) interpolation transformation calculation, (4) skeleton modification, and (5) object reconstruction. The inputs to the algorithm are two binary images,  $I_A$  and  $I_B$ , and the number of slices,  $L$ , to produce between them. In the context of correcting automatic segmentations from 3D EM datasets,  $I_A$  and  $I_B$  represent segmentations of the feature of interest on image slices A and B, which are separated from one another by  $L$  axial increments. In the first step of the MSI process, the binary objects of  $I_A$  and  $I_B$  are skeletonized, producing the skeletons  $S_A$  and  $S_B$ , respectively. Skeletonization is performed using a distance transform-based method, such that the skeleton is a grayscale representation in which the intensity of each pixel corresponds to the value of its Euclidean distance transform. Such skeletons have the advantage of maintaining information about image scale in addition to translation and rotation. Following skeleton generation, the transform,  $T_{AB}$ , required to bring the skeletons  $S_A$  and  $S_B$  into registry with one another is determined. In their report of the MSI algorithm, Chatzis and Pitas used the iterative closest point (ICP) algorithm to furnish this transform (Besl and McKay, 1992; Zhang, 1994).  $T_{AB}$  is then scaled to accommodate the number of slices specified by  $L$ , generating  $L$  intermediate transforms which are subsequently applied to  $S_A$  to yield skeletons of all interpolated slices between  $I_A$  and  $I_B$ . In the final step, binary objects are reconstructed from each interpolated skeleton by applying the inverse distance transform in which all skeletal points are treated as centers of maximal binary disks.

In the MATLAB implementation of MSI presented here, a number of modifications to the original algorithm have been made. First, skeletons are generated using the discrete curve evolution (DCE) algorithm of Bai and colleagues, which has been shown to preserve object topology in the presence of significant shape variations better than classical skeletonization algorithms (Bai et al., 2007). A second improvement in this implementation of MSI is the use of the coherent point drift (CPD) algorithm for the nonrigid point set registration of the skeletons  $S_A$  and  $S_B$  (Myronenko and Song, 2010). This algorithm was demonstrated to be more robust than ICP, even in the presence of noise, for a variety of complex shapes. Finally, a second mode of operation involving the registration and transformation of object perimeters, rather than skeletons, was implemented. Both modes, skeletonization and perimeterization, are available in the code presented here.

The process is initiated using the MATLAB script *msi3d\_dce\_cpd.m* (Appendix C.3.5), which requires the user to supply  $I_A$ ,  $I_B$ ,  $L$ , and the desired mode (skeletonization or perimeterization). Skeletonization is performed using a MATLAB implementation of the DCE skeletonization algorithm that was downloaded from the author's website (<https://sites.google.com/site/xiangbai>). The MATLAB functions *im2cpd.m* (Appendix C.3.6) and *cpd2im.m* (Appendix C.3.7) are used to convert image coordinates to the format needed for CPD and vice versa. Point set registration is performed using a MATLAB implementation of the CPD algorithm downloaded from the author's personal website (<https://sites.google.com/site/myronenko/research/cpd>). Skeleton transformation is performed in the main script, and object reconstruction from these skeletons is performed by the function *skel2obj.m* (Appendix C.3.8). When the perimeterization mode is selected, interpolated objects are filled using the function *perimFill.m* (Appendix C.3.9). If the lateral dimensions of  $I_A$  and  $I_B$  are  $M \times N$ , the final output of this process is an image matrix of size  $M \times N \times (L+2)$  that contains both  $I_A$  and  $I_B$  as well as all interpolated slices

between them. A MATLAB script, *msi3d\_display.m* (Appendix C.3.10) was created to provide easy visualization of results as well as track the scaling, translational, and rotational changes that occur from slice-to-slice.

To test the algorithm, three scripts were written to produce example input images for use as  $I_A$  and  $I_B$ . These scripts, *genCircleTest.m*, *genSquareTest.m*, and *genArbitraryTest.m* (Appendix C.3.11-C.3.13), generate images in the form of circles, squares, and arbitrary binary objects, respectively. Furthermore, they allow for the user-specified translation and rotation of  $I_A$  and  $I_B$ , providing a robust test for the CPD-based registration of interpolated slices. Figures 3.4-3.7 illustrate the use of these scripts to create test images and show the output interpolations generated by the proposed MSI process. These examples demonstrate the robustness of the process; accurate results were obtained for translated and scaled circles (Figures 3.4 and 3.5) and differently scaled, translated, and rotated squares (Figure 3.6) and nuclear profiles (Figure 3.7). The validity of the interpolations are confirmed by demonstrating the consistency of their translations, rotations, and scalings from slice to slice.

An example use case for this algorithm is illustrated in Figures 3.8 and 3.9. Shown in Figure 3.8 are eight consecutive slices through an automatic segmentation of a nucleus generated using the method described in Chapter 1. By overlaying the binary segmentations on the original image, it is clear that slices 2-5 contain clumps of false positive pixels at the top of the nuclear profile. Though such false positives would not significantly impact the results of nuclear detection or centroid localization, they are extremely detrimental if accurate nuclear morphologies are desired, and would typically require manual removal. However, it is demonstrated that these errors can be automatically and reliably removed using the novel implementation of the MSI algorithm described here. In Figure 3.9, the automatic segmentations of slices 2-5 have been

removed and replaced by interslice interpolations generated using slice 1 as  $I_A$ , slice 6 as  $I_B$ , and a value of four for  $L$ . Such interpolations yield results that are much more faithful to the correct nuclear morphology. A method for automatically detecting such erroneous automatic segmentations will allow MSI to be applied immediately after automatic segmentation, drastically reducing the manual correction time a human would need to invest. The development of such a method will be the goal of future work.

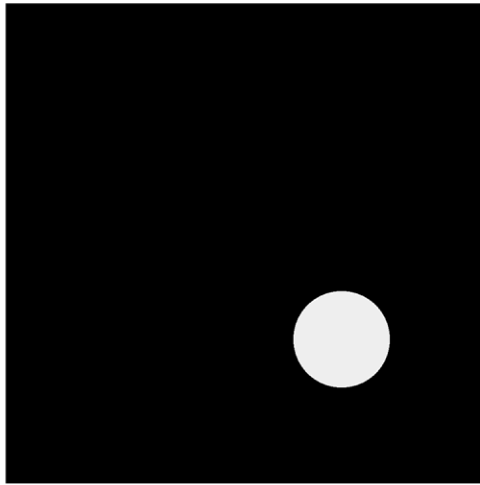
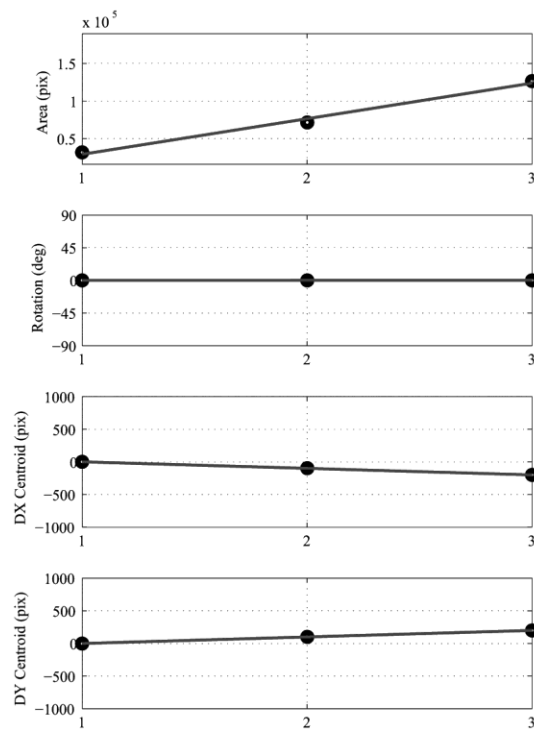
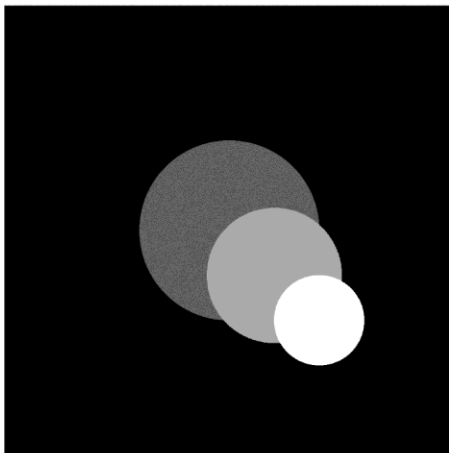
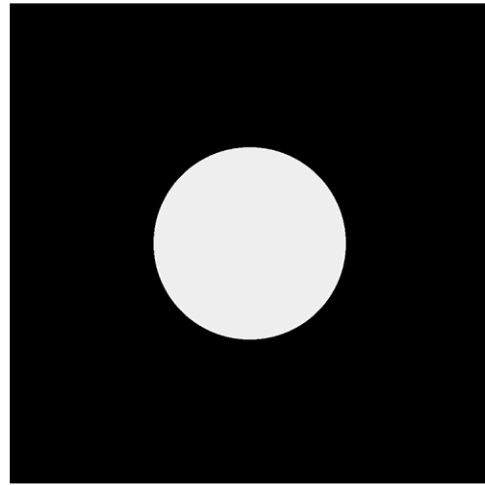
### 3.2.3. Contour and mesh generation

The result of automatic segmentation is a stack of  $N$  binary images in the TIF format with the same dimensions as the input dataset. In order to visualize and accurately quantify organelle morphologies, these binary images must be converted to 3D meshes. Using the IMOD software package, this would traditionally be performed in a three-step process consisting of the following steps: (1) append all individual TIF files to a single MRC stack, (2) create contours around each 2D connected component, and (3) generate meshes using the contours of all 3D connected components. However, the second step in this workflow is extremely rate-limiting for certain organelles. For example, in a typical dataset of size 32,000 x 24,000 x 1,500, each 2D segmentation would likely contain many thousands of mitochondrial cross-sections. Using a single CPU, contour generation for such a dataset using the program *imodauto* takes on the order of days.

To expedite this process, a workflow for its parallelization was developed and implemented. A flowchart of this entire workflow is shown in Figure 3.10. Parallelization is achieved by the submission of SGE array jobs on the NBCR cluster, rocce.ucsd.edu. All processes are invoked by the wrapper script *contourgen.sh* (Appendix C.3.14). At a minimum, this script requires (1) the path containing the series of segmented TIF images, (2) an output path, and (3) either the location of the original MRC stack or the origin and

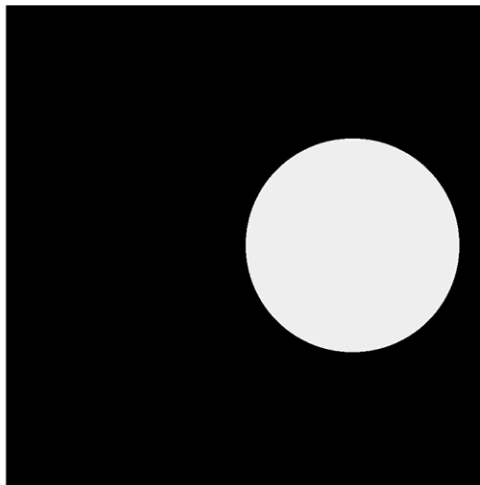
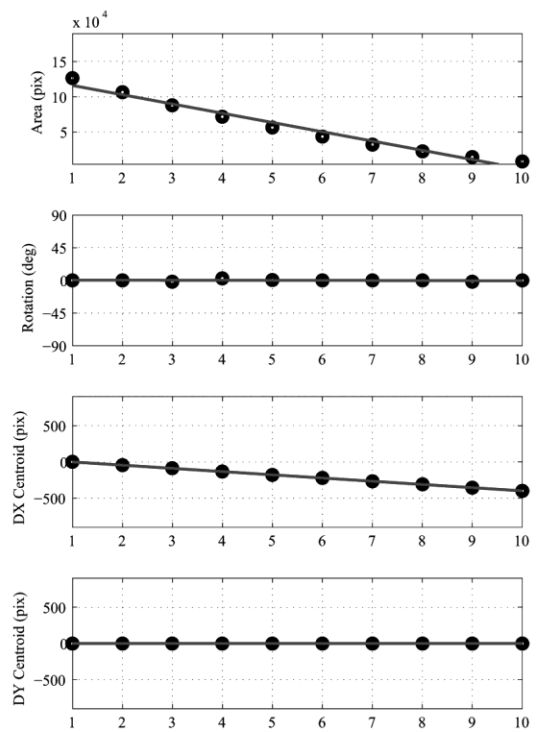
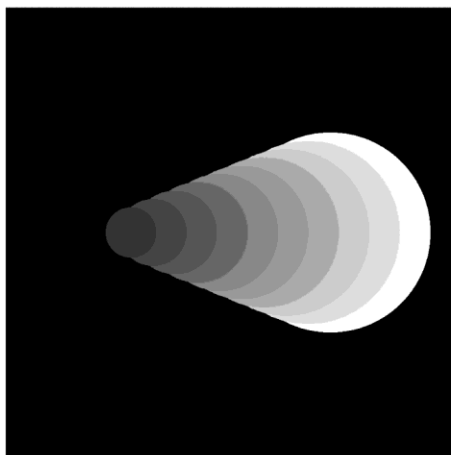
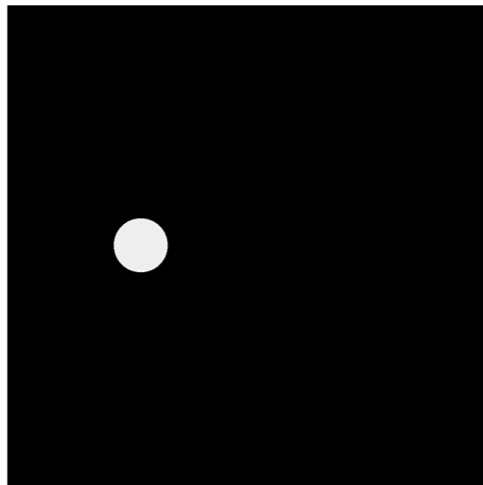
**Figure 3.4. A single interslice interpolation between two differently scaled and translated circles.** A single slice ( $L = 1$ ) was interpolated between two differently scaled circles that were translated with respect to one another. The first circle has a radius of 100 pixels ( $I_A$ ) while the second circle has a radius of 200 pixels and is translated by 200 pixels away from the center of the image in both X and Y ( $I_B$ ). The results of interslice interpolation using the MSI algorithm are shown (middle left). The plots on the middle right indicate the area, rotation, change in X centroid, and change in Y centroid of each slice. Values for rotation and centroid position are considered to be zero for  $I_A$ . A solid line on each plot indicates the linear regression formed by all points. The inputs to the MATLAB scripts used to produce these data are displayed at the bottom of the image. Interpolation was performed using perimeterization since it is faster than interpolation by skeletonization and the enhanced accuracy afforded by skeletonization was not necessary for simple shapes such as circles.



$I_A$  $I_B$ 

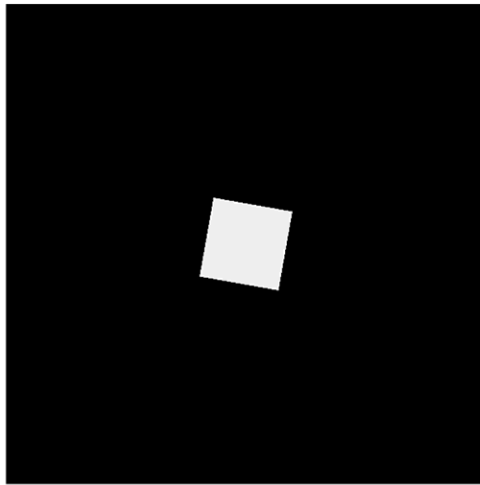
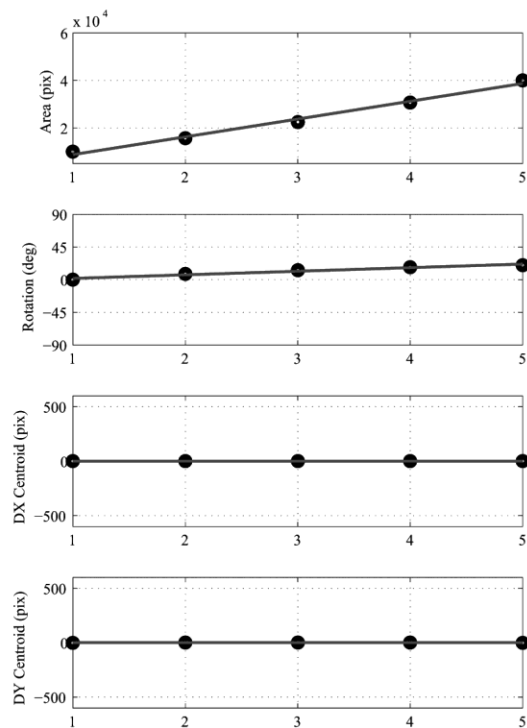
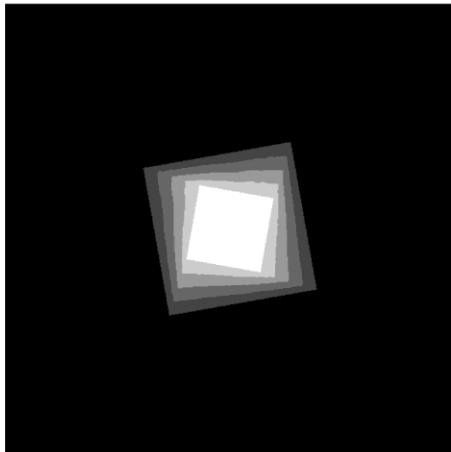
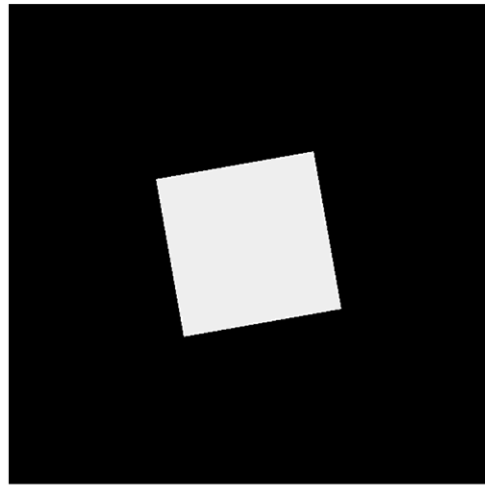
```
>> [IA,IB] = genCircleTest(100,200,[200 200]);
>> interp = msi3d_dce_cpd(IA,IB,1,2,0,1,1);
>> msi3d_display(interp,'interp1',0);
```

**Figure 3.5. Multiple interslice interpolations between two differently scaled and translated circles.** Eight slices ( $L = 8$ ) were interpolated between two differently scaled circles that were translated with respect to one another. The first circle has a radius of 200 pixels and is translated by +200 pixels from the center of the image in X ( $I_A$ ). The second circle has a radius of 50 pixels and is translated by -200 pixels from the center of the image in X ( $I_B$ ). The results of interslice interpolation using the MSI algorithm show consistent scaling and translation across all eight interpolated slices, as indicated visually as well as by the linearity of the changes in centroid location and area of each slice.

$I_A$  $I_B$ 

```
>> [IA,IB] = genCircleTest(200,50,[200 0],[-200 0]);
>> interp = msi3d_dce_cpd(IA,IB,8,2,0,1,1);
>> msi3d_display(interp,'interp2',0);
```

**Figure 3.6. Multiple interslice interpolations between two differently scaled and rotated squares.** Three slices ( $L = 3$ ) were interpolated between two differently scaled squares that were rotated with respect to one another. The first square has a dimension of 100 pixels and is rotated by  $-10^\circ$  from the horizontal ( $I_A$ ). The second square has a dimension of 200 pixels and is rotated by  $+10^\circ$  from the horizontal ( $I_B$ ). The results of interslice interpolation using the MSI algorithm demonstrate consistent scaling and rotation across all interpolated slices. Interpolation was performed using skeletonization, since the improved accuracy was helpful when interpolating the linear edges present in squares.

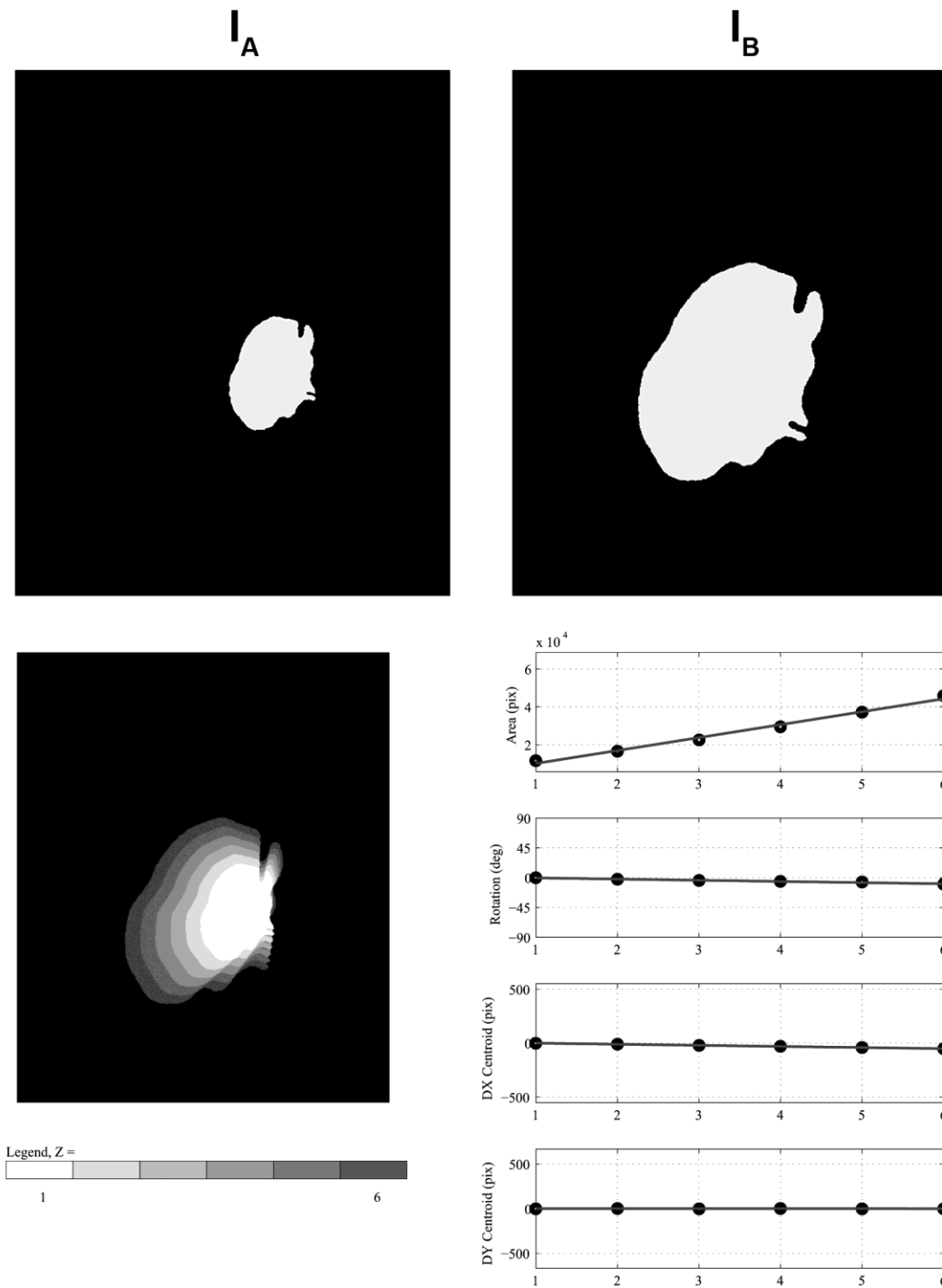
$I_A$  $I_B$ 

```

>> [IA,IB] = genSquareTest(100,200,[0 0],[0 0],[-10,10]);
>> interp = msi3d_dce_cpd(IA,IB,3,1,0,1,1);
>> msi3d_display(interp,'interp2',0);

```

**Figure 3.7. Multiple interslice interpolations between two differently scaled and rotated segmentations of an invaginated nucleus.** Four slices ( $L = 4$ ) were interpolated between two differently scaled automatic segmentations of an invaginated SCN nucleus that were also rotated with respect to one another. The first nucleus is scaled by a factor of 0.5 and rotated by  $+5^\circ$  from the horizontal ( $I_A$ ). The second nucleus was left at its original size and rotated by  $-5^\circ$  from the horizontal ( $I_B$ ). The results of interslice interpolation using the MSI algorithm demonstrate consistent scaling and rotation across all interpolated slices. Interpolation was performed using skeletonization. Importantly, interslice interpolations generated by transforming skeletons maintain their membrane topology; invaginations remain present and properly scaled in all interpolated slices.



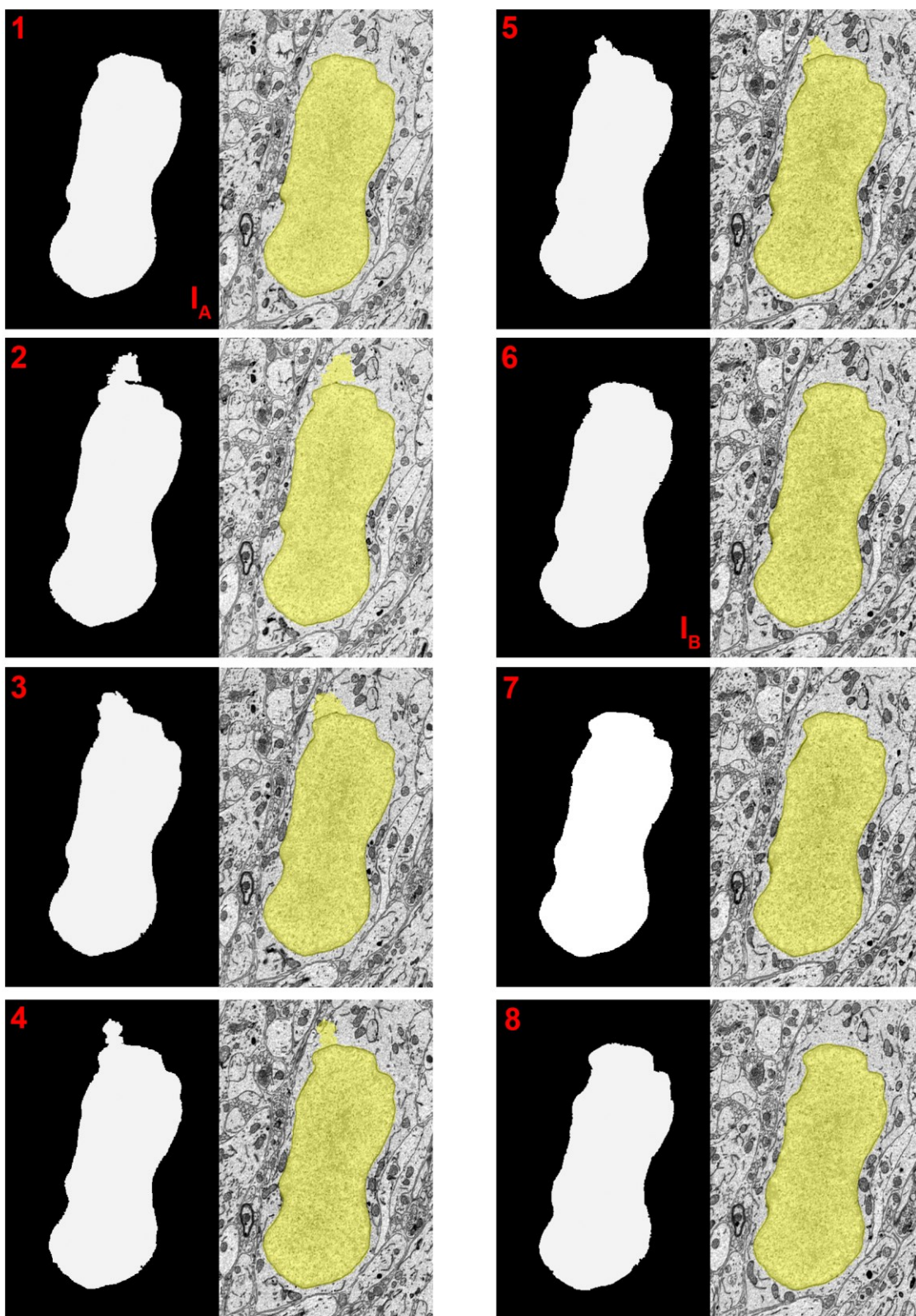
```

>> [IA,IB] = genArbitraryTest('nucleus.tif',0.5,1,[50 50],[0 50],5,-5);
>> interp = msi3d_dce_cpd(IA,IB,4,1,0,1,1);
>> msi3d_display(interp,'interp4',0);

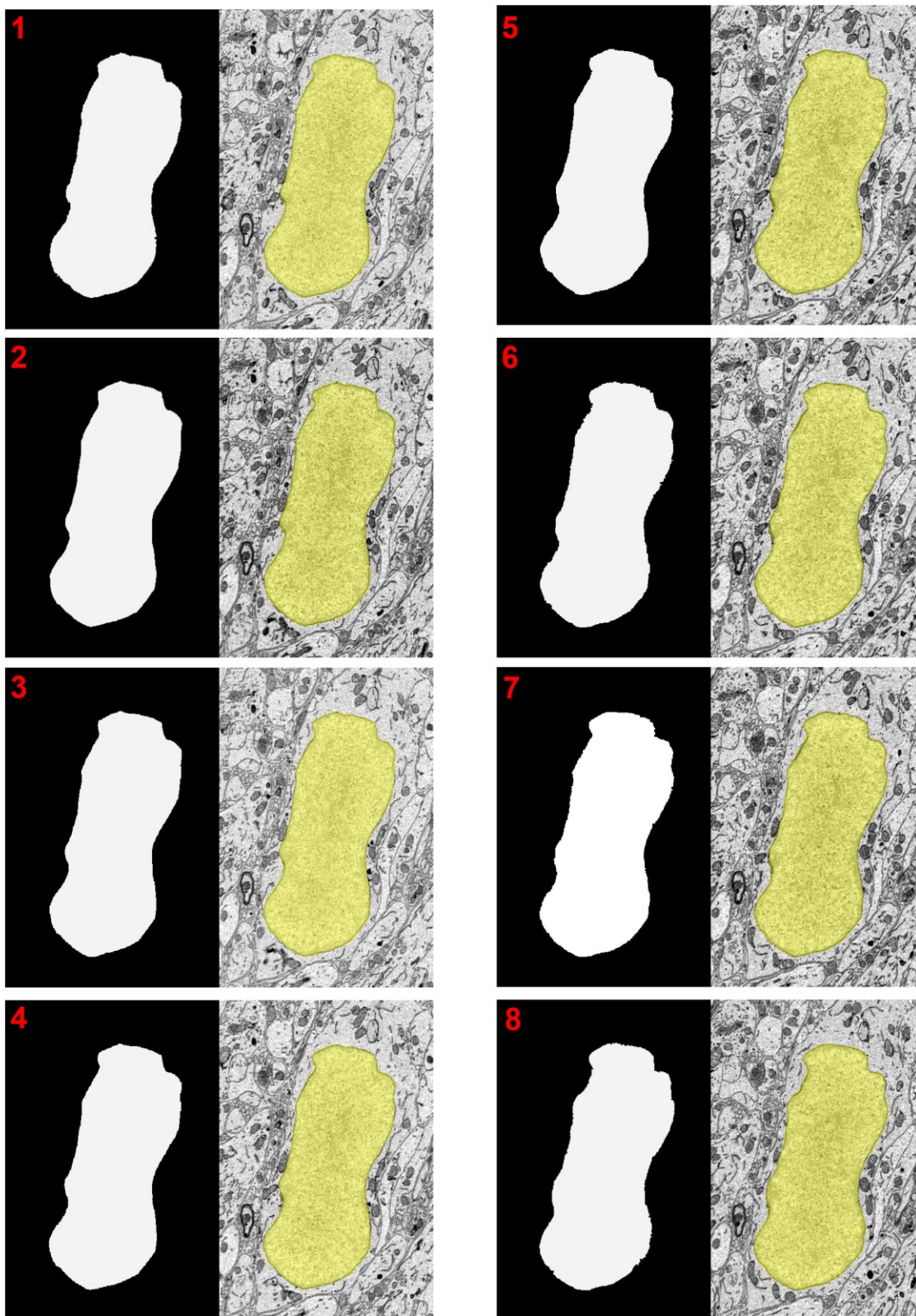
```

**Figure 3.8. An example scenario in which automatic segmentation accuracy benefits from post-processing by interslice interpolation.** Depicted here are eight slices from the automatic segmentation of a single nucleus generated using the method described in Chapter 2. Shown to the right of each automatic segmentation is its overlay on the original image. As can frequently occur with automatically generated segmentations, some clusters of false positive pixels are present at the top of the nuclear profiles in slices 2-5. Such artifacts have a deleterious effect when quantifying nuclear morphologies, such as volume, surface area, and membrane curvature. Using interslice interpolation, these poorly segmented slices can be rejected and replaced with more accurate interslice interpolations using the segmentation of slice #1 as  $I_A$ , the segmentation of slice #6 as  $I_B$ , and a value of four for  $L$ . The results of this interpolation is shown in Figure 3.7.





**Figure 3.9. Replacing poorly segmented slices with interslice interpolations increases morphological accuracy.** Depicted here are the same eight slices from Figure 3.6, but with the inaccurate segmentations of slices 2-5 replaced by their interslice interpolations generated using the MSI code presented here. The accuracy of these interpolated segmentations is demonstrated by visualizing their overlay on the original image data. Meshes generated from these interpolations would be significantly more representative of the true nuclear morphology.



pixel spacing specified by the header of the original MRC stack. Optionally, the user can also specify smoothing parameters to be applied during contour generation.

In the first step, *contourgen.sh* submits an array job using the SGE submission script *tif2mod2D.q* (Appendix C.3.15). All images contained in the segmentation stack are processed in parallel, and a sequence of operations are performed on each image. In the first step, the image is converted to the MRC format using the IMOD program *tif2mrc*. The header of this new, single-slice MRC file is then edited to reflect the origin and pixel spacing of the original MRC stack using the IMOD program *alterheader*. Contours are then generated around each 2D connected component using the IMOD program *imodauto*, which generates one, single-slice model file for each image. The program *imodtrans* is then used to translate each model file to its proper axial location. The number of contours,  $N_C$ , contained in each model file is determined by parsing an ASCII representation of the model, obtained by the program *imodinfo*. This number is then written to a separate text file for use by subsequent programs. The final outputs from this job script are: (1)  $N$  single-slice model files and (2)  $N$  ASCII files specifying the number of contours contained in each model file.

When all of these individual model files are joined together using the program *imodjoin*, each slice is given its own object, resulting in a final model with  $N$  objects. This scenario does not allow for accurate meshing using *imodmesh*, which will not make connections across contours of different objects. Thus, to ensure proper meshing, all contours need to be joined into a single object. The next step of the workflow achieves this via submission of an array job using the SGE script *mod2point2D.q* (Appendix C.3.16). For each single-slice model file, this script runs the MATLAB function *mod2point2D.m* (Appendix C.3.17), which uses the MatTomo (<http://bio3d.colorado.edu/imod/matlab.html>) library to read the IMOD model file binary and output the desired point listings. First, each

model file is converted to an ASCII representation of point listings compatible with the IMOD program *point2model*. In this representation, each line contains five space-delimited integer values, formatted as follows:

Object\_number Contour\_number X\_coordinate Y\_Coordinate Z\_coordinate

The object number is one for all slices and contours. The starting contour number,  $C_0$ , for each model file is determined by summing the number of contours contained in the ASCII files from all previous slices according to the following formula, in which  $i$  represents the current slice number:

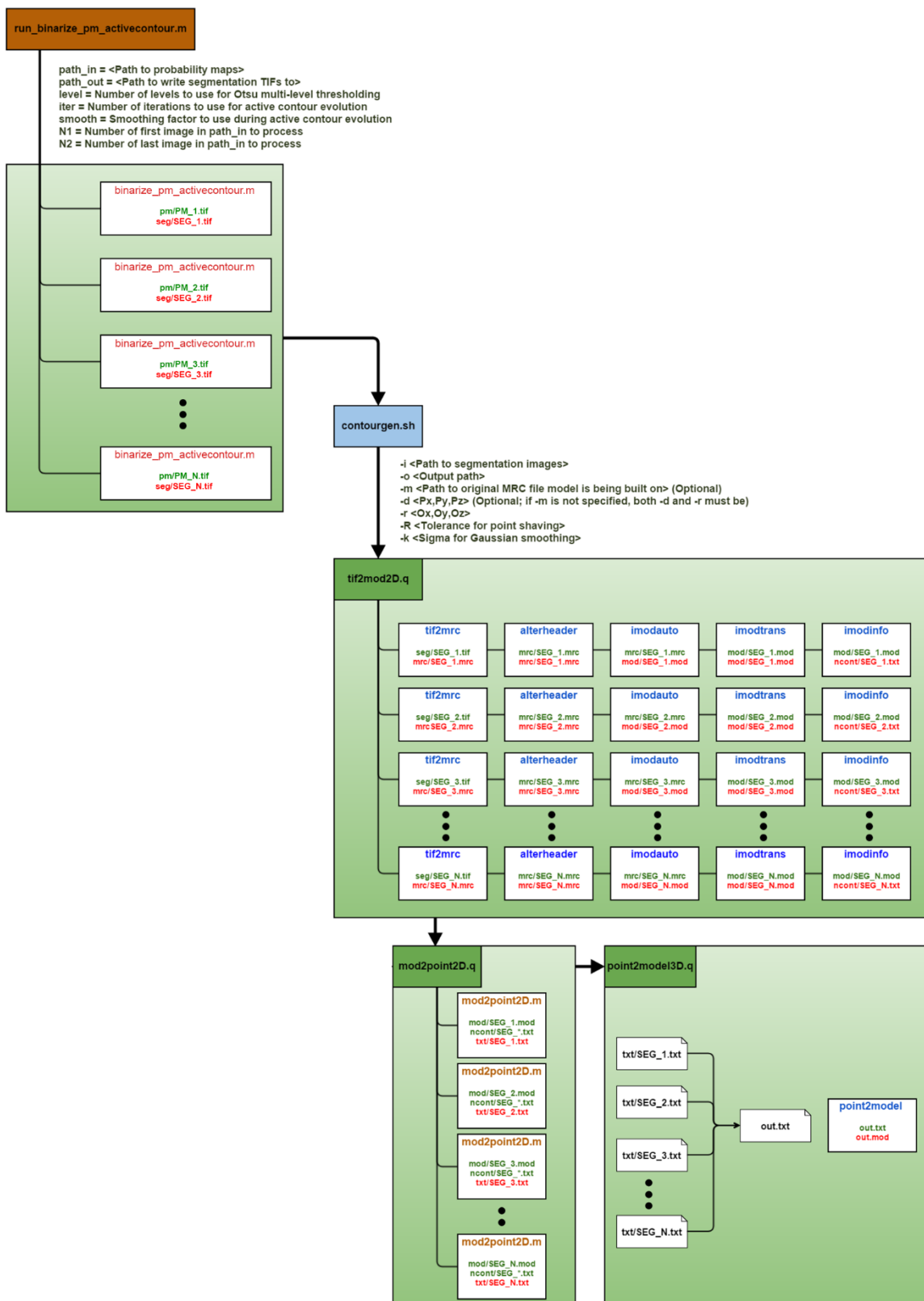
$$C_0^i = 1 + \sum_{j=1}^{i-1} N_C^j$$

All subsequent contours within each file are numbered sequentially. The final output from this process is a series of  $N$  ASCII files specifying the coordinates of every point on each slice.

In the final step, all  $N$  ASCII files are appended to one single point listing file, which is then converted to a model file using the IMOD program *point2model*. This is performed using the script *point2mod3D.q* (Appendix C.3.18). Thus, the final output of the entire workflow is a model file with one object that consists of contours around all 2D connected components in the entire segmented stack. This model file consists of one object containing contours around all 2D connected components. All 3D connected components are then meshed together using the program *imodmesh*. Individual 3D components are then sorted into separate objects using the IMOD program *sortsurf*. These individual objects are then remeshed following separation to yield the final, automatically generated, and dataset-wide organelle surface renderings. Examples of dataset-wide surface renderings generated using this method are depicted in Figures 3.11-3.13.

**Figure 3.10. A flowchart of the steps involved in contour and mesh generation from large-scale automatic segmentations.** Segmentations are generated from probability maps using binarization with automatically seeded active contours, as previously described. The result of this step is a stack of N binary images the size of the original stack (SEG\_1.tif, ..., SEG\_N.tif). Following segmentation, the script *contourgen.sh* is run, which parallelizes contour generation by submitting SGE array jobs. In the first step, each segmented image is converted to the MRC format, then edited so that the MRC header information matches that of the original stack. IMOD contours are then generated around each 2D connected component using the IMOD program *imodauto*.





### 3.2.4. Tools for the automatic analysis of nuclear morphology

#### 3.2.4.1. Nuclear and nucleolar morphology and positioning

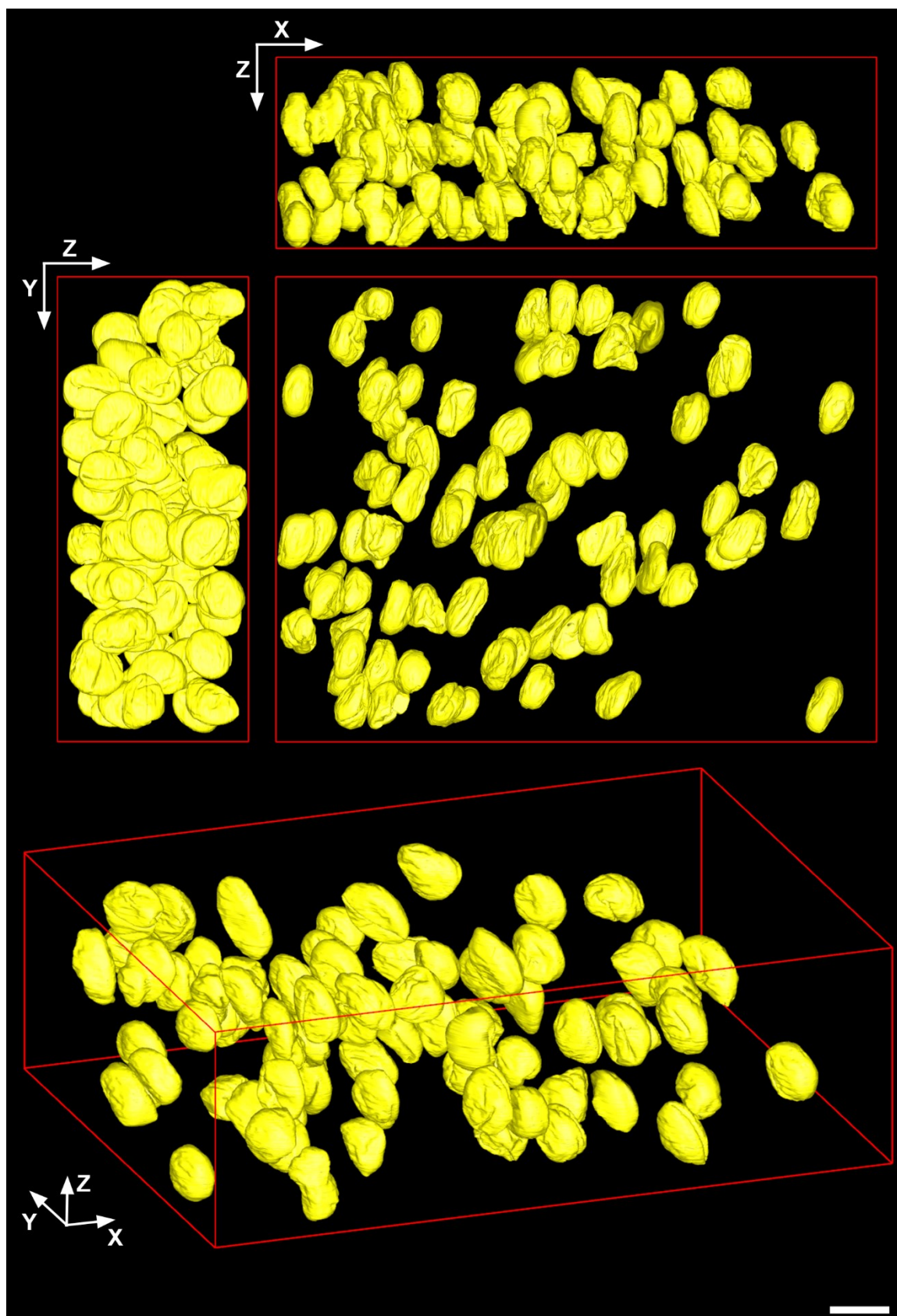
To facilitate rapid analysis, a workflow for the automatic reporting of nuclear metrics and statistics was developed. This process is initiated by the script *sbem\_analyze\_nuclei.sh* (Appendix C.3.19), which uses the dataset-wide surface renderings (Figures 3.11-3.13) generated using the methods of the previous section as input. The script outputs metrics on nuclei and nucleoli if model files containing segmentations of both are provided; alternatively, it can output statistics of only one or the other if desired. The rest of this section will focus on the case in which both model files have been provided.

The first task is to automatically group the segmented nucleoli with their corresponding nucleus. This is done by analyzing the centroids of the nucleoli and determining which centroids fall within the bounding box of a given nucleus. A loop is run over all nucleolus objects, and the centroid of each nucleolus is output to a temporary text file using the program *imodinfo*. After this, a loop is run over all nucleus objects. For each nucleus, the coordinates of the eight points specifying the corners of its 3D bounding box are extracted using the program *imodinfo*. All nucleoli whose centroids fall within the bounds of this box are assigned to the given nucleus. A new IMOD model is initiated, and the objects corresponding to these nucleoli are then joined to the object corresponding to the nucleus. At the end of this loop,  $N_N$  new model files will have been produced, where  $N_N$  is the number of nucleus objects in the input, dataset-wide model file.

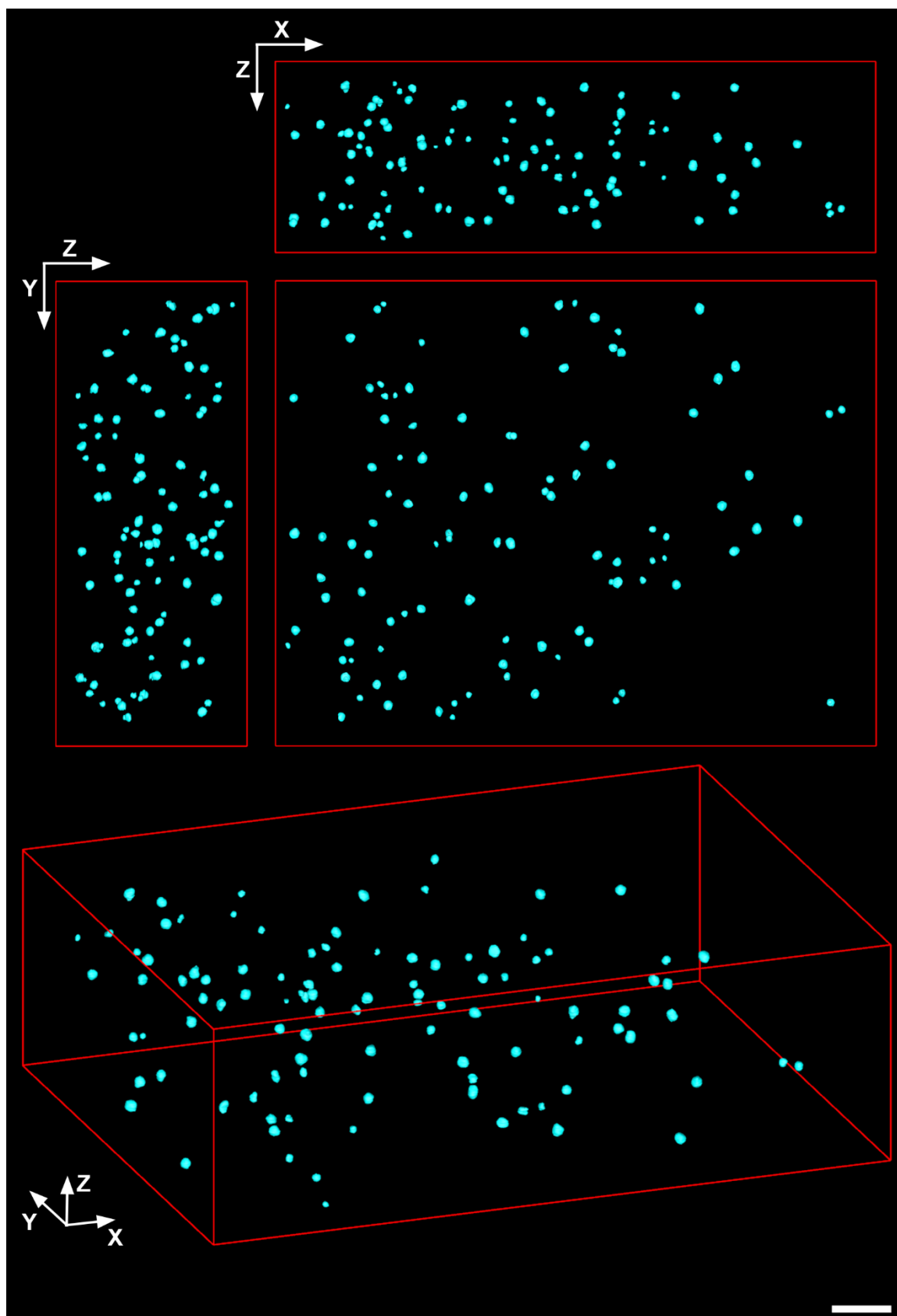
In the next step, volume and surface area are computed for each nucleus and nucleolus and output to corresponding ASCII files. A loop is run over each of the newly generated model files. For each file, the nuclear volume and surface area are computed using the program *imodinfo*, and the nuclear surface area to volume ratio is then calculated



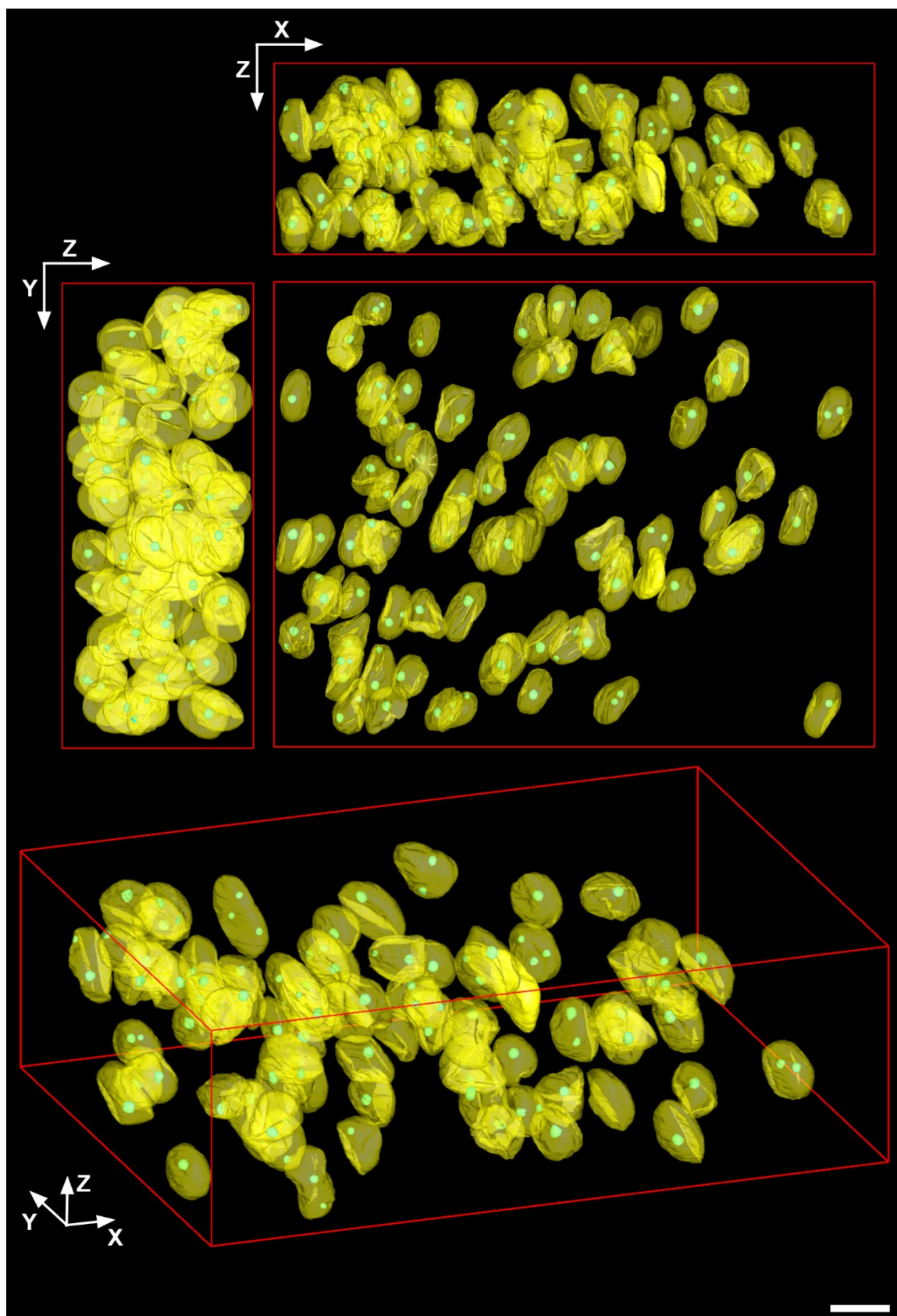
**Figure 3.11. Automatically generated surface renderings of nuclei from the ZT04 SCN SBEM dataset.** Multiple views of surface renderings generated from automatic segmentations are shown here. The top figures depict the renderings from orthogonal views in the XY, XZ, and YZ planes. The bottom figure depicts the renderings in a rotated volume to illustrate depth. Eighty-one total nuclei were included in this representation, and most nuclei are heavily invaginated (scale bar = 10  $\mu\text{m}$ ).



**Figure 3.12. Automatically generated surface renderings of nucleoli from the ZT04 SCN SBEM dataset.** Multiple views of surface renderings generated from automatic segmentations are shown here. The top figures depict the renderings from orthogonal views in the XY, XZ, and YZ planes. The bottom figure depicts the renderings in a rotated volume to illustrate depth (scale bar = 10  $\mu\text{m}$ ).



**Figure 3.13. Combined renderings of automatically segmented nuclei and nucleoli from the ZT04 SCN SBEM dataset.** Shown here are different views of the combination of renderings of nucleoli (cyan) and all 81 nuclear surface renderings (yellow, translucent). From this visualization, it is clear that while most nuclei contain a single nucleolus, there are many instances of cells with multiple nucleoli (scale bar = 10  $\mu\text{m}$ ).



from these two values. These three metrics are written to a single line of the output ASCII file *nucleus\_morphology.txt* such that each line of the file corresponds to the metrics of the *i*th nucleus. The file is written in a comma-separated values (CSV) format to allow easy import to analytical programs such as Microsoft Excel. The contents of this file for the current test SCN dataset (CCDBID: 81739) are given in Appendix D.1. Within the same loop, the individual volumes of all nucleoli belonging to the *i*th nucleus are computed and written to the output CSV file *nucleolus\_morphology.txt*. In addition, the total number of nucleoli, the total nucleolar volume, and the nucleolar volume fraction for the current nucleus are written to this file. The contents of this file for the current dataset are given in Appendix D.2. A graphical depiction of these values for a single nucleus as well as screenshots of the output CSV files are shown in Figure 3.14.

Following the extraction of morphological parameters, metrics pertaining to nucleolar positioning within the nucleus are computed. The positions of all nuclear and nucleolar centroids are calculated, and the distances between each nucleolar centroid and its corresponding nuclear centroid are computed using the IMOD program *mtk*. These distance values are output to the CSV file *dist\_centroid.txt*, and the contents of this file for the current dataset are given in Appendix D.3. A graphical representation of these distances is given in Figure 3.15. Next, the distances between the surfaces of each nucleolus and its corresponding nucleus are calculated using *mtk*. These values are output to the CSV file *dist\_nuclear\_envelope.txt*. The contents of this file for the current dataset are given in Appendix D.4., and a graphical representation of these distances is given in Figure 3.16. Definitions for all computed metrics are given in Table 3.2.

**Table 3.2. The metrics automatically computed and output during the morphological analysis of nuclei.** Parameters are computed using the script *sbem\_analyze\_nuclei.sh* and output to CSV files to enable further analysis. The metrics for which no equation is given are computed directly from IMOD programs. For all metrics: N = nucleus, NL = nucleolus, C = centroid, NE = nuclear envelope, i = nuclear index, j = nucleolar index.

Metric	Variable	Equation
Nuclear volume	$V_i^N$	N/A
Nuclear surface area	$S_i^N$	N/A
Nuclear surface area to volume ratio	$SVR_i^N$	$\frac{S_i^N}{V_i^N}$
Number of nucleoli	$N_i^{NL}$	N/A
Nucleolar volume	$V_{i,j}^{NL}$	N/A
Total nucleolar volume	$V_i^{NL}$	$\sum_{j=1}^{N_i^{NL}} V_{i,j}^{NL}$
Nucleolar volume fraction	$f_i^{NL}$	$\frac{V_i^{NL}}{V_i^N}$
Distance between nucleolar and nuclear centroids	$d_{i,j}^{NL-C}$	N/A
Distance between nucleolar and nuclear surfaces	$d_{i,j}^{NL-NE}$	N/A

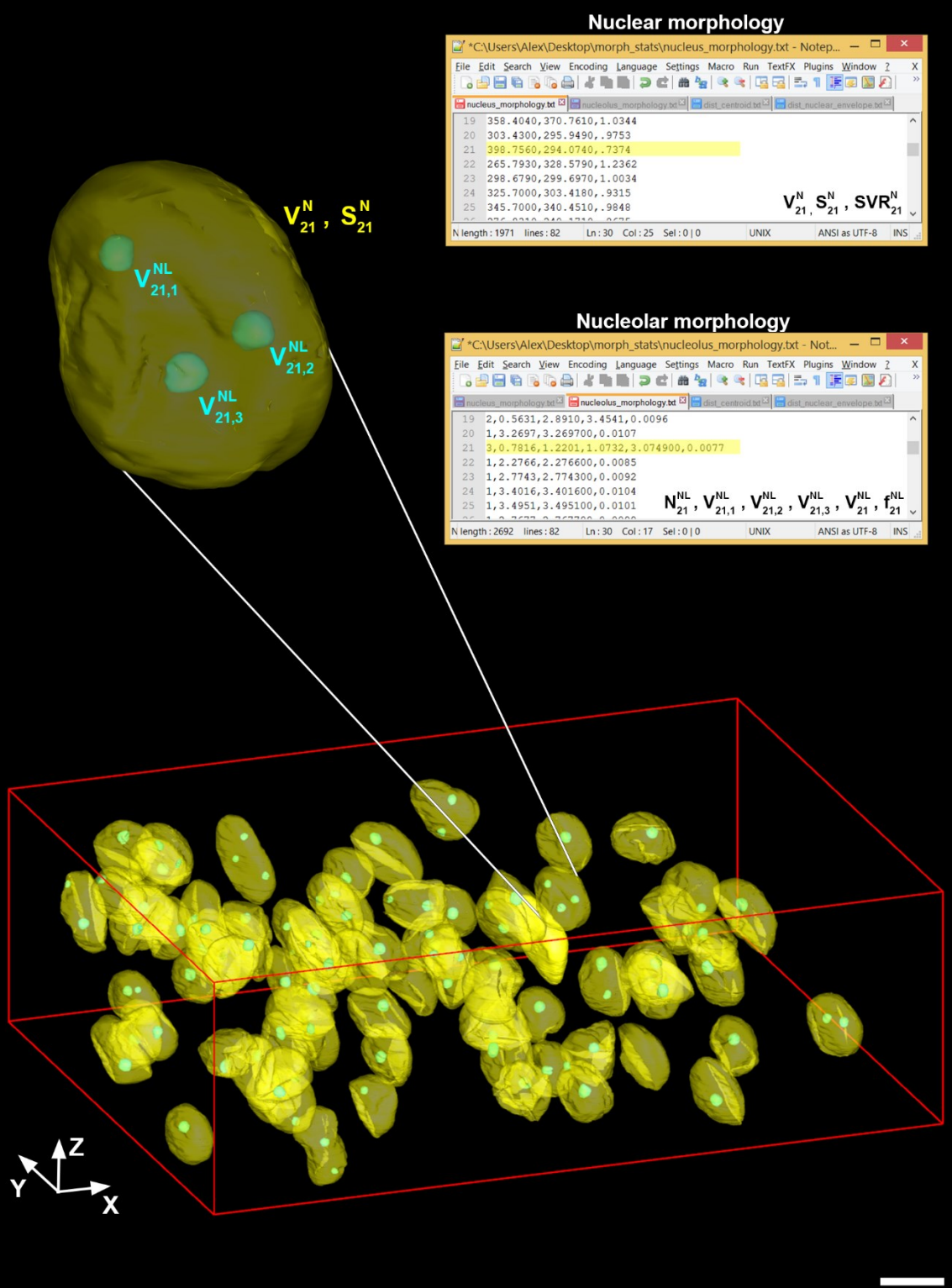
### 3.2.4.2. Advanced metrics for characterizing nuclear topology

In this section, a number of metrics for characterizing nuclear topology and membrane folding are proposed. The validity of each of these metrics was ascertained by comparison to qualitatively assigned degrees of nuclear folding. The most intuitive and widely used metric for this quantification is the SVR, which was automatically computed using the methods of the previous section. However, preliminary observations suggested



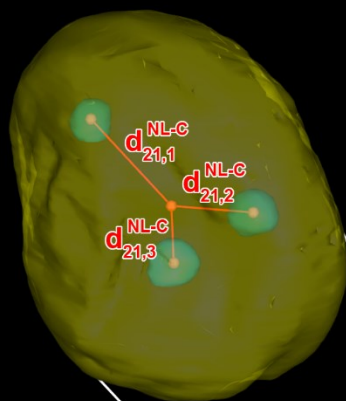
**Figure 3.14. An example of the nuclear morphological characterization workflow.** Shown at the bottom of the figure is the same combined nuclear and nucleolar surface rendering depicted in Figure 3.13. The model file specifying this rendering was supplied as input to the morphological characterization workflow described here. A single nucleus from this rendering (nucleus #21) is magnified to highlight the morphological parameters calculated in the first step of the workflow, namely nuclear and nucleolar volumes and surface areas. Screenshots of the CSV files generated by the workflow are shown to the right. The row of each CSV file pertaining to the nucleus shown is highlighted in yellow to show correspondence. (Bottom scale bar = 10  $\mu\text{m}$ )

## Nucleus #21



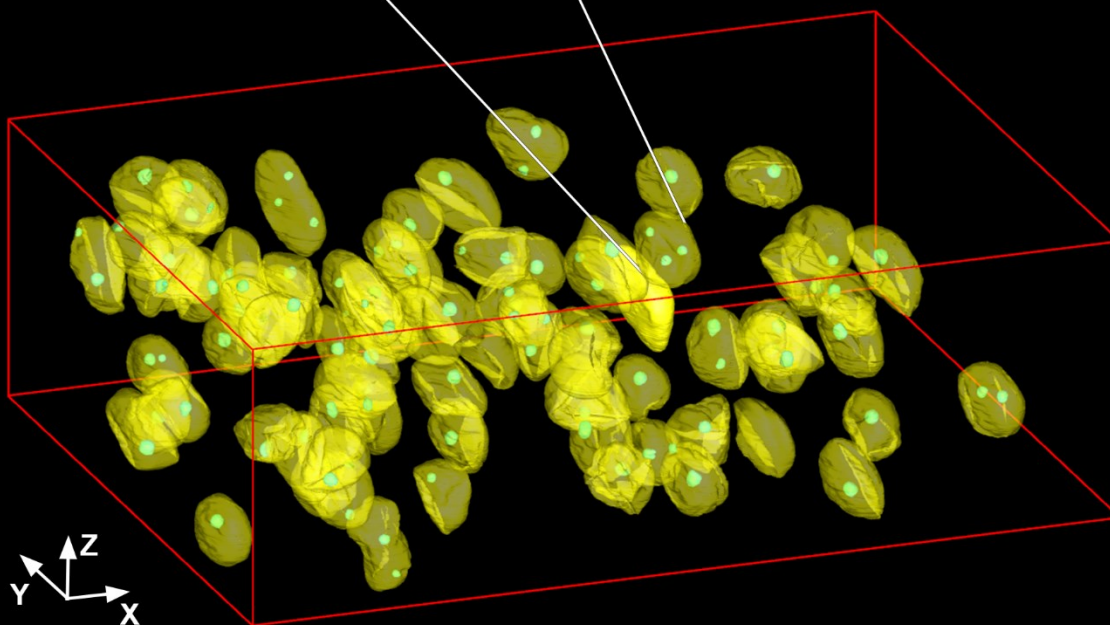
**Figure 3.15. An example of the nucleolar positioning workflow.** The same nucleus depicted in Figure 3.14 is shown here. The centroids of each nucleolus and the nucleus are denoted by red spheres. The distances between the nuclear centroid and each nucleolar centroid are indicated by red lines and the corresponding variable. Each of these distances was calculated using the proposed workflow, and the results are shown in a screenshot of the CSV generated by the script. The row pertaining to the current nucleus is highlighted in yellow. (Scale bar = 10  $\mu\text{m}$ )

## Nucleus #21

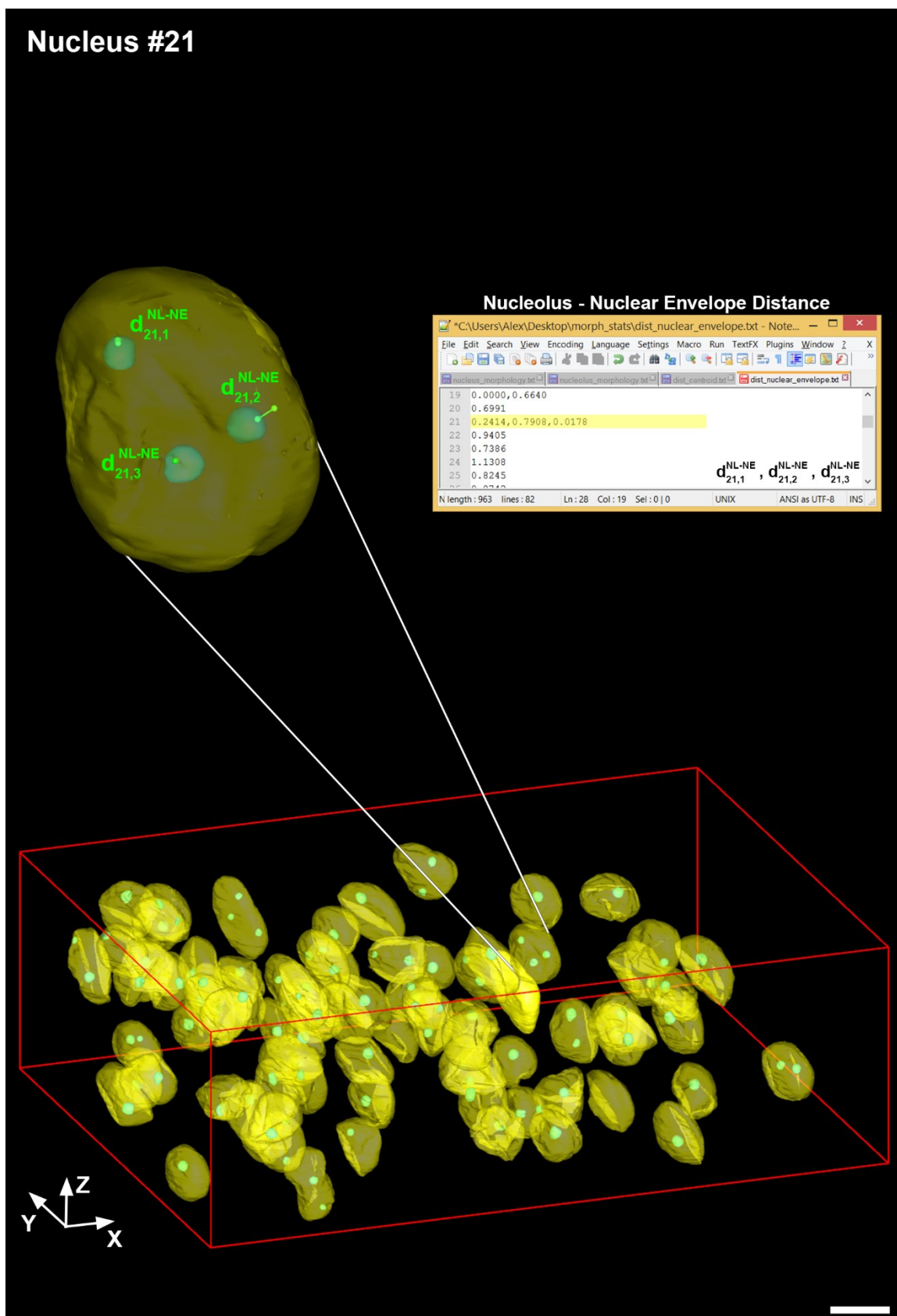


## Nucleolus - Nuclear Centroid Distance

```
*C:\Users\Alex\Desktop\morph_stats\dist_centroid.txt - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window 2 X
nucleus_morphology.txt nucleus_morphology.txt dist_centroid.txt dist_nuclear_envelope.txt
19 10.4327, 3.8781
20 1.5227
21 8.3864, 4.2999, 5.1474
22 4.0229
23 3.0527
24 3.0628
25 2.8466
N length: 970 lines: 82 Ln: 28 Col: 22 Sel: 0 | 0 UNIX ANSI as UTF-8 INS
```



**Figure 3.16. An example of the nucleolar positioning workflow, Continued.** The same nucleus depicted in Figures 3.14 and 3.15 is shown here. The minimum distance between the surfaces of each nucleolus and the nuclear envelope were computed using the IMOD program *mtk*. These distances are indicated here by green lines, and the points of contact on each surface are denoted by green spheres. The results are shown in a screenshot of the CSV generated by the script. The row pertaining to the current nucleus is highlighted in yellow. (Scale bar = 10  $\mu\text{m}$ )



that the SVR alone did not appear to be an adequate descriptor for the degree of folding seen in the nuclei of this dataset. Therefore, three other metrics were developed, and their validity for quantifying the degree of nuclear invagination was ascertained. The first of these metrics, the 3D invagination factor ( $IF_{3D}$ ), is an expansion of the 2D IF proposed by Lafarga and colleagues that was previously introduced in Chapter 3.1.1 (Lafarga et al., 1992). In their formulation of the IF, the ratio of perimeter to cross-sectional area was normalized such that it yielded a value of one for a perfect circle, and values greater than one when any folds were present. This formula was expanded to 3D by considering the ratio of surface area to volume rather than that of perimeter to area. The same normalization was implemented, such that the ratio of surface area (S) to volume (V) produces a value of one for a perfect sphere. The formula that yields such a normalization is:

$$IF_{3D} = \frac{S}{V^{2/3}} \times K_{3D}$$

$$K_{3D} = 0.25 \left( \frac{16}{9\pi} \right)^{1/3}$$

The  $IF_{3D}$  can be calculated directly from the automatically generated results from the previous section.

The second metric, the convex hull difference (CHD), is the difference between the volume of the nuclear convex hull (CH) and the volume of the nucleus:

$$CHD_i^N = V_i^{CH,N} - V_i^N$$

Nuclear CH computation is initiated using the Bash wrapper script *sbem\_convexHull.sh* (Appendix C.3.20). This script works on each single-nucleus model file output by the methods of the previous section. The CH is computed using the MATLAB script *sbem\_convexHull.m* (Appendix C.3.21), which calculates the 3D CH from the Delaunay triangulation of the points comprising the nuclear boundary. The output CH is saved as a

series of images, which are then converted to an IMOD model file using the program *imodauto*. The volume of the CH is subsequently computed using the program *imodinfo*. Examples of the CHs generated using this method are shown in Figures 3.17 and 3.18.

The final metric considered here is the nuclear shape index,  $\sigma$ . First introduced by Koenderink and van Doorn, the shape index is a single, angular measure that describes local shape and surface topology as a function of the two principal curvatures,  $K_{\max}$  and  $K_{\min}$ . Possible values for the shape index range from -1 to +1, where negative values indicate a more convex, or cup-like, local topology, and positive values imply a more concave, or cap-like, local topology (Koenderink and van Doorn, 1992). The local shape index is given by the following:

$$\sigma = \frac{2}{\pi} \arctan \frac{K_{\max} + K_{\min}}{K_{\max} - K_{\min}}$$

Values of the shape index are computed locally by calculating the principal curvatures at each triangle vertex of the mesh. Nuclear meshes generated using the previous methods are first converted from the IMOD model file format to the VRML format using the IMOD program *imod2vrml*. The VRML object is imported to Amira and then remeshed to optimize the distribution of vertices. The principal curvatures at each vertex are determined, and the local shape index at each vertex is calculated according to the formula given above. The surface integral of the shape index over the nuclear surface is given by:

$$\sigma_i^N = \frac{2}{\pi} \int_S \arctan \frac{K_{\max} + K_{\min}}{K_{\max} - K_{\min}} dA$$

This integral is computed by exporting the vertex positions and shape index scalar field to a custom MATLAB script. The integral is then normalized by dividing by the nuclear surface area. The final output of this process is a final value, the total nuclear shape index, which ranges from -1 to +1. A number closer to -1 indicates that the nucleus is more



heavily invaginated, while a number closer to +1 indicates a smoother, more spherical nucleus. Shape index calculation is initiated using the wrapper script *sbem\_totalCurvature.sh* (Appendix C.3.22), which handles file conversions and launches Amira. The Amira Tcl script *sbem\_totalCurvature.hx* (Appendix C.3.23) is then launched, which performs remeshing, shape index calculation, and vertex and scalar field output. It also generates movie representations featuring 360° rotations of the nucleus about two axes in both the MPEG and GIF formats. Integration is performed with the MATLAB script *totalCurvature.m* (Appendix C.3.24). Meshed representations of nuclei overlaid with their local shape index scalar fields are given in Figure 3.19-3.21.

After these metrics were calculated for all 81 nuclei, the goal was to determine which of them gave the most faithful correspondence to qualitatively observed degrees of nuclear folding. Each nucleus was assigned a numerical value of 0-3 to indicate its qualitative degree of invagination, where a score of zero represented no folding and a score of three represented a heavily folded nucleus. Examples of nuclei assigned values of 1, 2, and 3 are shown in in Figures 3.19, 3.20, and 3.21, respectively. The values of SVR,  $IF_{3D}$ , CHD, and  $\sigma$  belonging to each score were grouped together, and the groups of each metric were compared to one another by means of multiple one-way analysis of variance (ANOVA) tests with Tukey's post-hoc tests for multiple comparisons (Table 3.3; Figure 3.22). The null hypothesis for each test was that the population means of each score were the same.

For the SVR, there was a statistically significant difference between score groups ( $F = 8.920$ ). A Tukey post-hoc test revealed significant differences between groups 0-2 ( $p < 0.01$ ), 0-3 ( $p < 0.0001$ ), and 1-3 ( $p < 0.01$ ). There were no statistically significant differences between other groups. Therefore, the SVR is unable to unambiguously differentiate between small qualitative differences in folding (i.e. between groups 0-1, 1-2,

**Table 3.3. The shape index and convex hull difference are able to discern small qualitative differences in nuclear invagination.** Reported here are the results from multiclass comparisons via one-way ANOVAs with Tukey's post-hoc tests. The CHD yields statistically significant differences between all qualitative score groups. The SVR does not yield statistically significant differences between adjacent score groups (i.e. 0-1, 1-2, and 2-3).

	F	Class Comparisons					
		0-1	0-2	0-3	1-2	1-3	2-3
SVR	8.920	ns	p < 0.01	p < 0.0001	ns	p < 0.01	ns
IF	11.64	ns	p < 0.01	p < 0.0001	ns	p < 0.001	p < 0.05
$\sigma$	14.60	p < 0.01	p < 0.0001	p < 0.0001	p < 0.05	p < 0.001	ns
CHD	17.59	p < 0.05	p < 0.0001	p < 0.0001	p < 0.05	p < 0.0001	p < 0.01

and 2-3). There was also a statistically significant difference between score groups for the IF<sub>3D</sub> (F = 11.64). However, similarly to the SVR, there were no statistically significant differences between groups 0-1 and 1-2 (Table 3.3). There was, however, a statistically significant difference between groups 2-3 (p < 0.05). The shape index and CHD provide much better metrics of nuclear invagination. For the shape index, statistically significant differences existed between all scoring groups except for the group 2-3. For the CHD, statistically significant differences existed between all scoring groups. The results of these comparisons are depicted graphically in Figure 3.22.

### 3.2.5. Delineation of individual neuronal compartments

Though nucleoli can be sorted into their corresponding cells based on overlap with nuclear bounding boxes, this same type of separation cannot be easily attained for cytoplasmic organelles such as mitochondria. Such accurate segregation is only possible if the cellular boundaries are previously demarcated. These cellular boundaries can then be used as binary masks to exclude all organelles that lie outside of them. Ideally, such segmentations would be obtained through one of the automatic methods discussed in Chapter 2.1.2, and such methods will be implemented in the future. Here, as a proof of

concept, neuronal membranes were manually segmented in their entirety throughout the dataset. Manual segmentation was performed in IMOD. Contours were traced on sections spaced a variable number of axial steps apart. The number of sections that could be skipped depended upon the feature being traced. When manually tracing a non-spiny neuronal cell body, as many as 5-10 slices could be skipped without significantly jeopardizing interslice accuracy. However, branched neurites, especially spiny dendrites, often required manually traced contours on every slice or every other slice. Following manual tracing, missing contours were filled in via an interslice interpolation algorithm. Each final segmentation was then meshed using the program *imodmesh* to provide a visual representation (Figure 3.23).

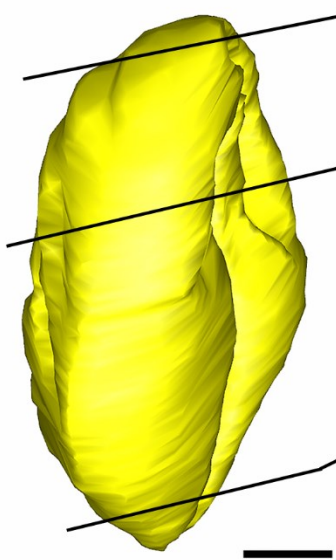
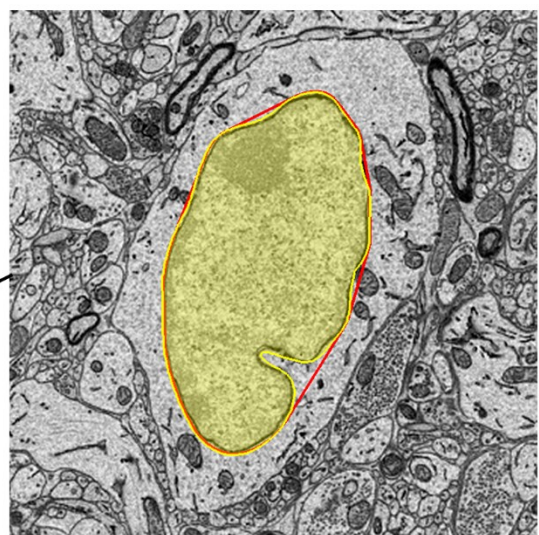
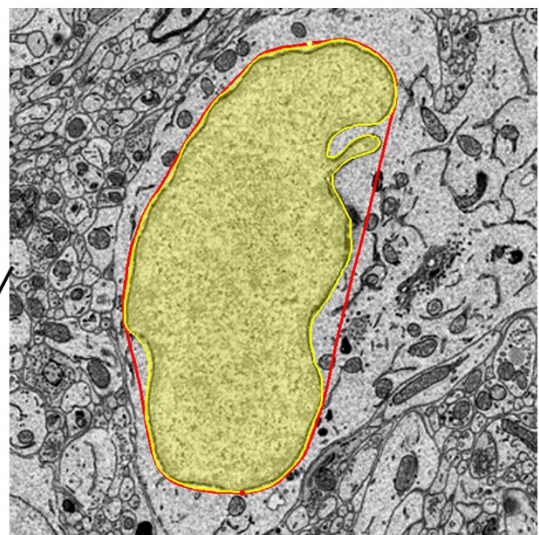
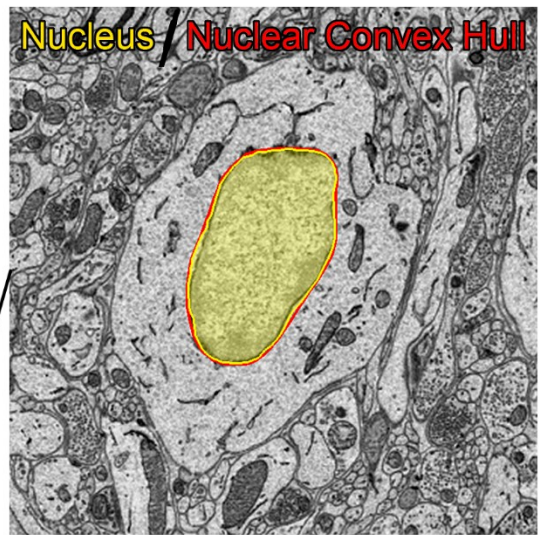
A stack of binary masks,  $B_{N,j}$ , where  $N$  indicates the index of the neuron and  $j$  indicates the slice number, were produced from each neuronal segmentation using the program *imodmop*. Mitochondria were automatically segmented from the same dataset by the methods described in Chapter 2, producing a stack of binary segmentations,  $S_j$ . Automatically segmented mitochondria were separated into their appropriate neuron by taking the slice-by-slice intersection of  $B_{N,j}$  and  $S_j$ , yielding the stack of neuron-specific mitochondrial masks,  $M_{N,j}$ :

$$M_{N,j} = B_{N,j} \cap S_j \quad \forall j \in \{1, \dots, N_{\text{slices}}\}$$

Surface renderings were then produced from these masked segmentations using the methods introduced in Chapter 3.2.3 (Figure 3.24). This method facilitates the cell-specific morphological and spatial analysis of automatically segmented organelles. A workflow for the automatic analysis of mitochondrial morphologies using such cell-specific models as input is currently being developed.

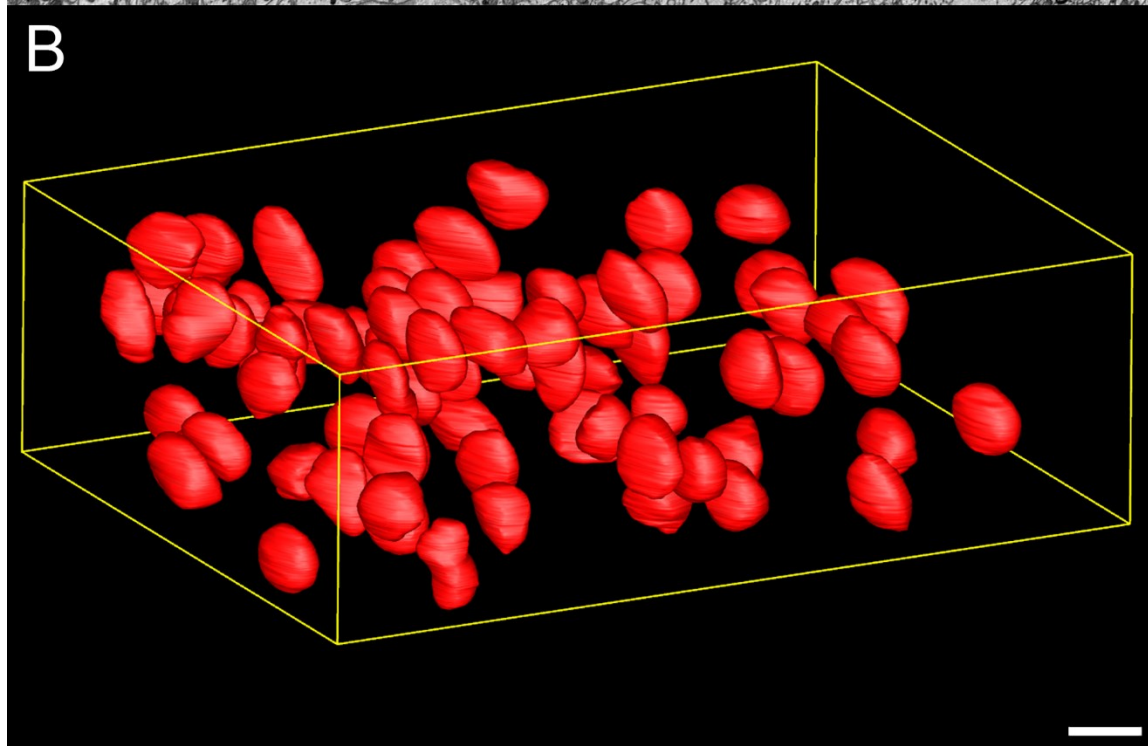
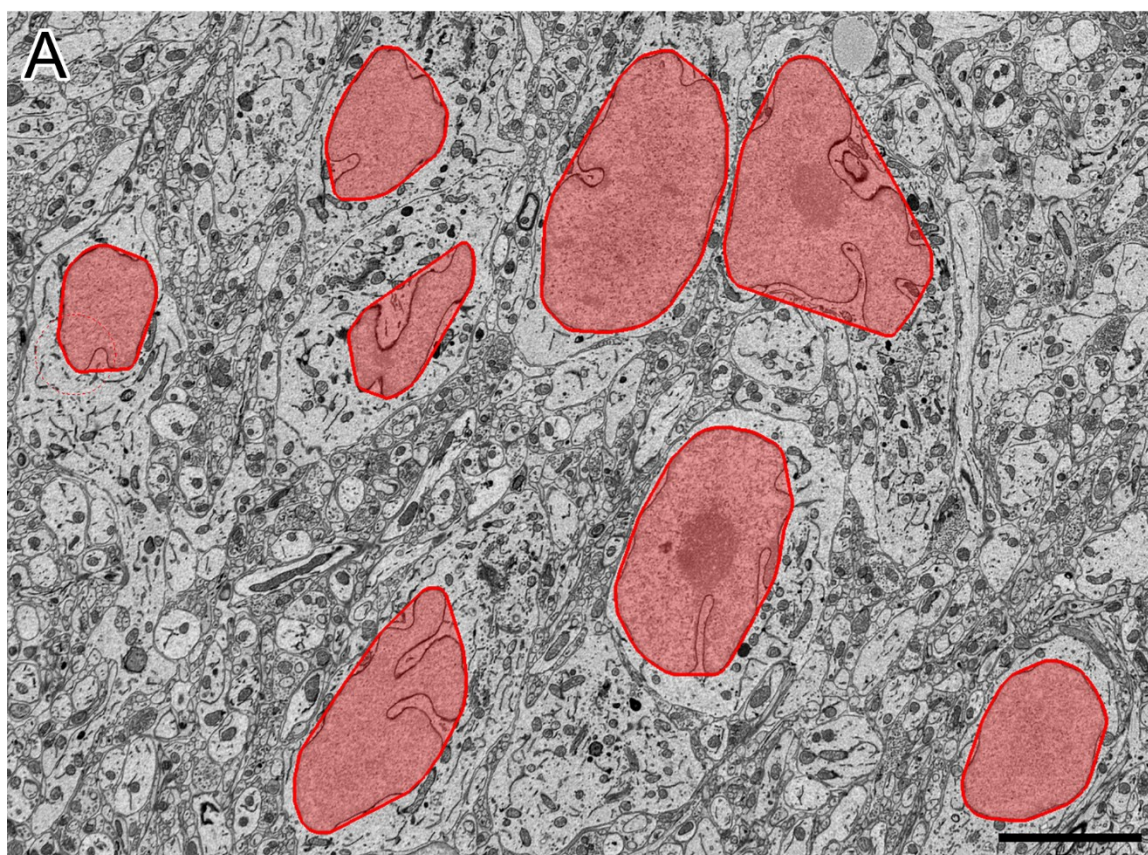
**Figure 3.17. The 3D convex hull for a single nucleus.** The automatically generated surface rendering of a single nucleus is shown in the lower left corner. Three 2D slices from this nucleus are depicted on the right, with overlays of the nuclear segmentation (translucent yellow) and the contour of the 3D convex hull (red). The convex hull imitates a rubber band being wrapped around the contour of the nucleus. The CHD can be visualized as the region between the CH and the nucleus, and is thus one of the more accurate descriptors of nuclear invagination. (Scale bar = 2  $\mu\text{m}$ )

Nucleus / Nuclear Convex Hull



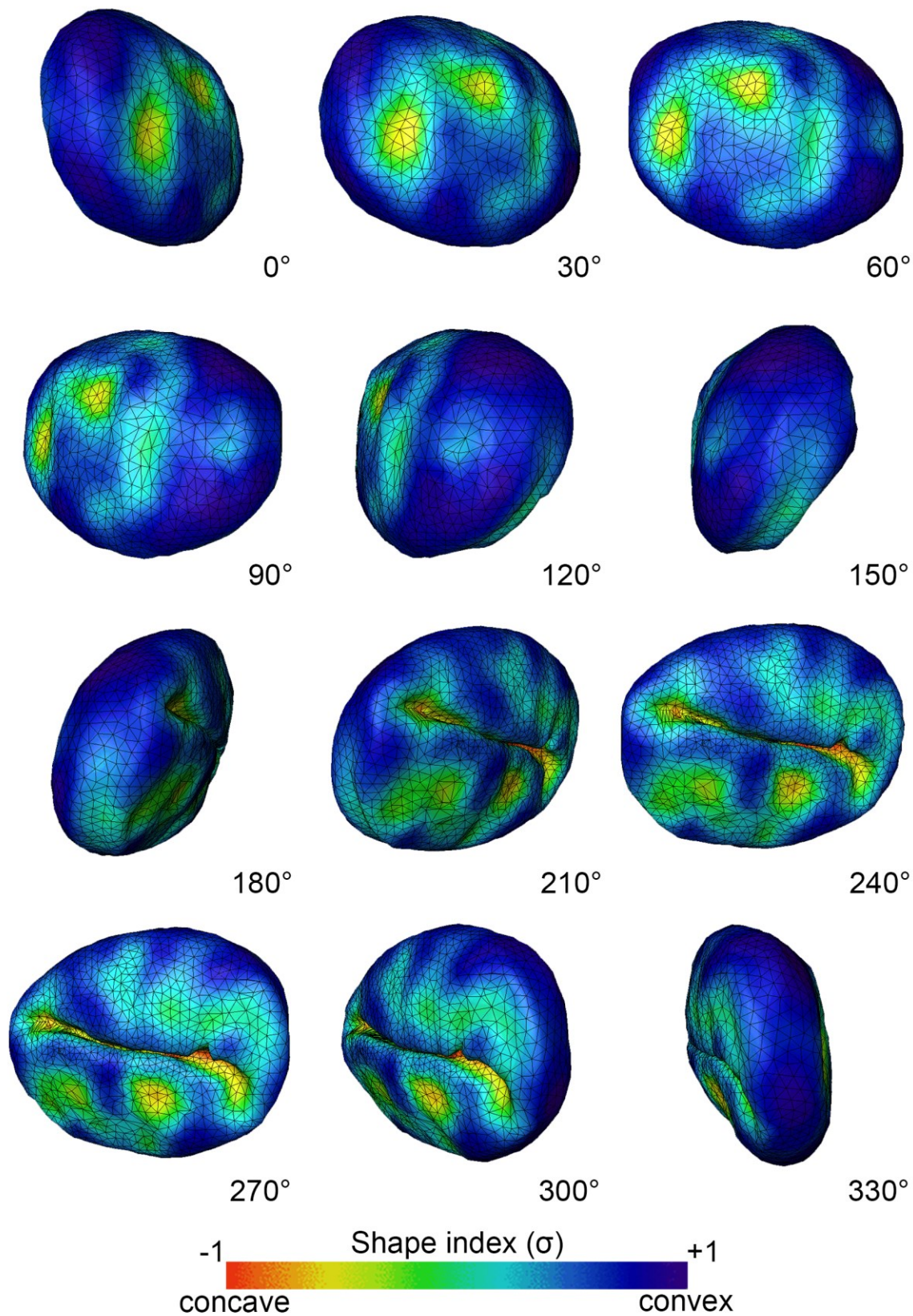
**Figure 3.18. 3D convex hull renderings for nuclei from a full SBEM volume.** 3D CHs were computed for all 81 nuclei depicted in Figures 3.14-3.16 using the method described here. A cross-sectional slice through the volume depicts eight accurately computed CHs overlaid on the original image data (A, scale bar = 5  $\mu\text{m}$ ). The volumes of each CH were computed directly from the surface renderings shown in B (scale bar = 10  $\mu\text{m}$ ).



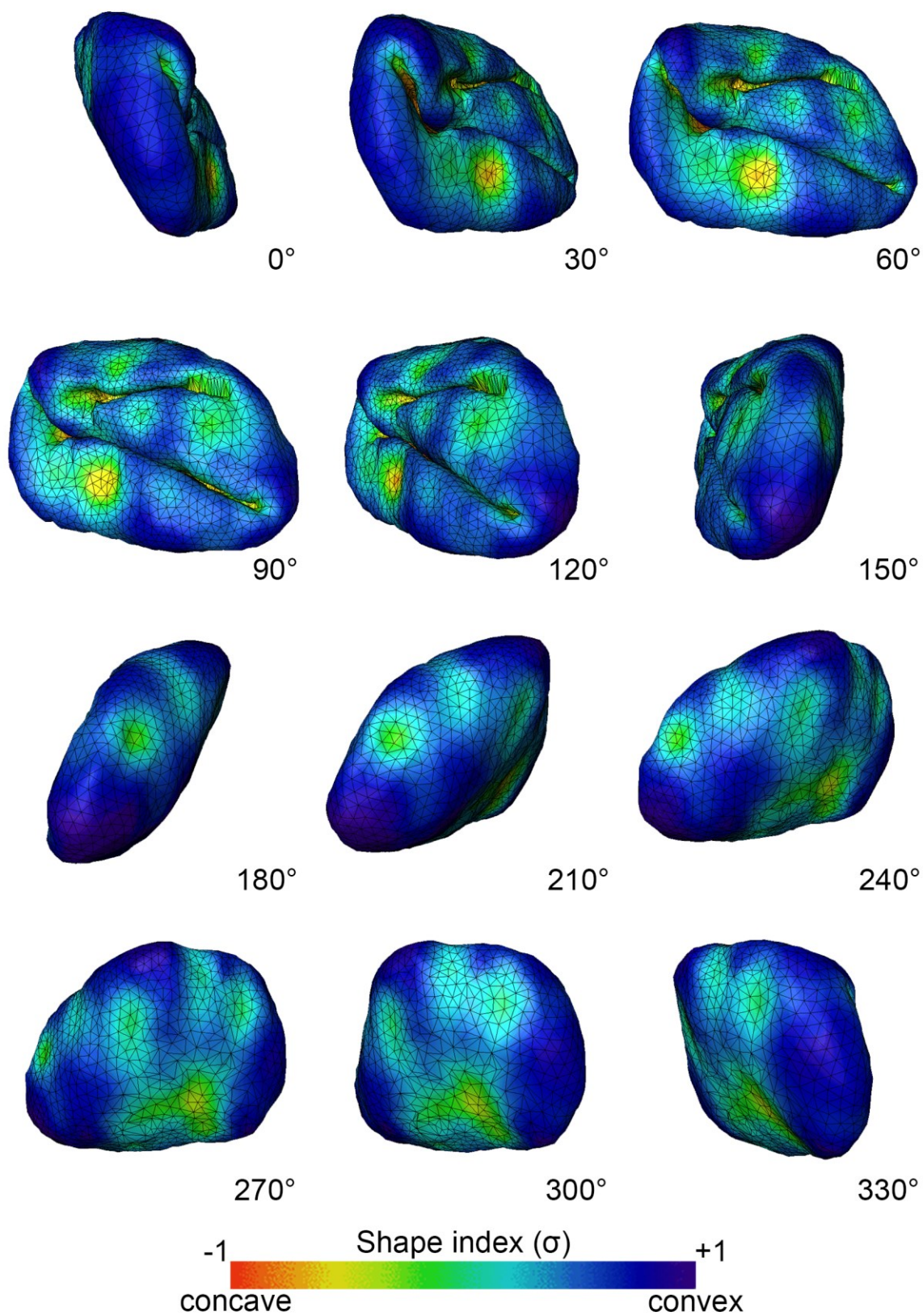


**Figure 3.19. The local shape index scalar field for a nucleus with a single invagination.** The local shape index was computed at each triangular vertex of the nuclear mesh using the method described in Chapter 3.2.4.2. Shown here are twelve views from the automatically generated movie of the local shape index scalar field overlaid on the mesh of a nucleus with a single, deep invagination. The degree of invagination of this nucleus was assigned a score of one on a scale of 0-3 by qualitative observation. Each image represents a successive rotation of  $30^\circ$  about the same axis of the nucleus. The shape index scalar field is coded to a colormap ranging from -1 to +1, as depicted at the bottom of the figure. Negative values indicate that the local shape is more concave, while positive values indicate a more convex local shape. As expected, the local shape index is more negative in the proximity of the invagination.





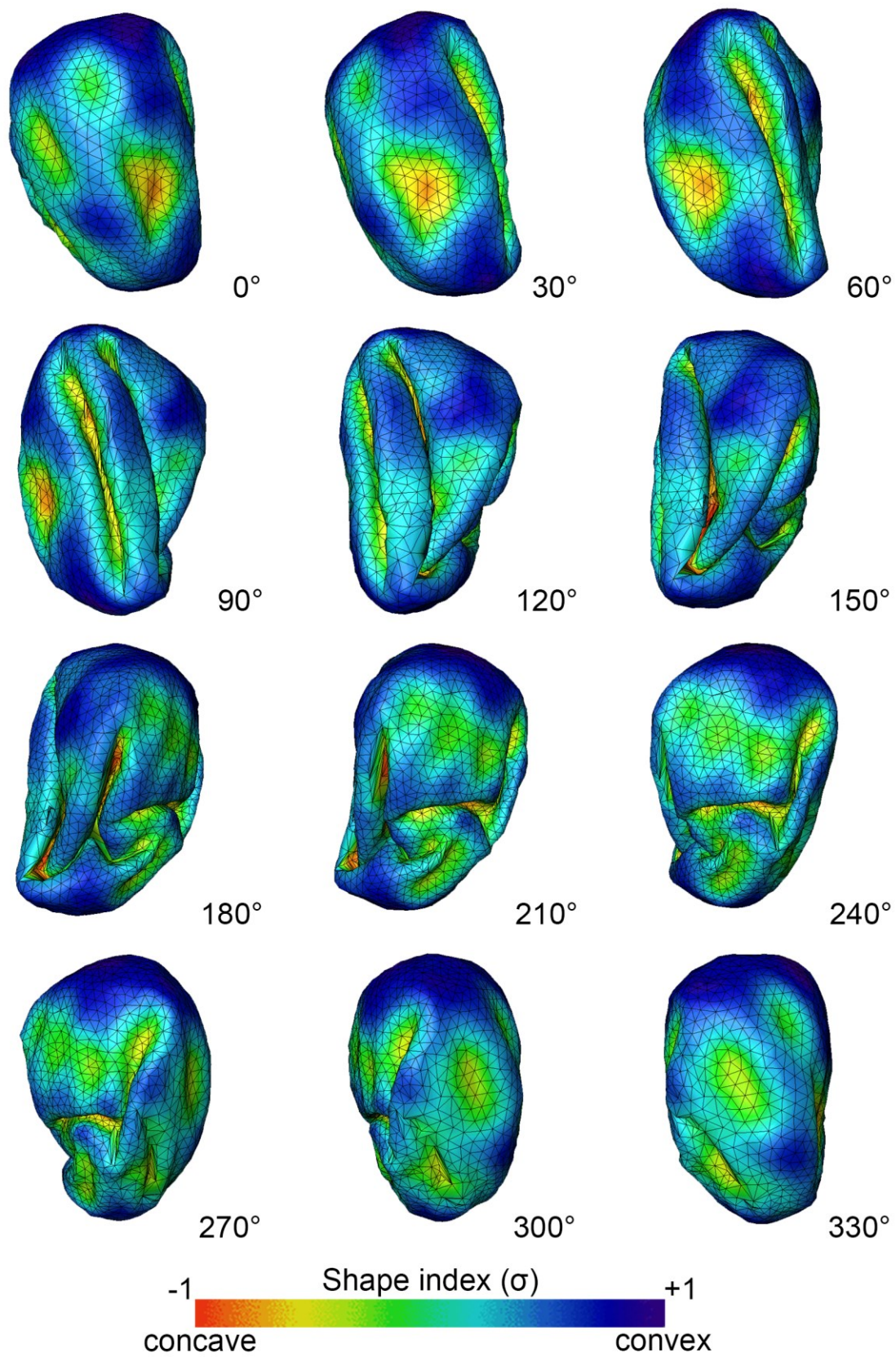
**Figure 3.20. The local shape index scalar field for a nucleus with multiple invaginations.** The local shape index was computed at each triangular vertex of the nuclear mesh using the method described in Chapter 3.2.4.2. Shown here are twelve views from the automatically generated movie of the local shape index scalar field overlaid on the mesh of a nucleus with multiple invaginations. The degree of invagination of this nucleus was assigned a score of two on a scale of 0-3 by qualitative observation. Each image represents a successive rotation of  $30^\circ$  about the same axis of the nucleus. The shape index scalar field is coded to a colormap ranging from -1 to +1, as depicted at the bottom of the figure. Negative values indicate that the local shape is more concave, while positive values indicate a more convex local shape. As expected, the local shape index is more negative in the proximity of invaginations.



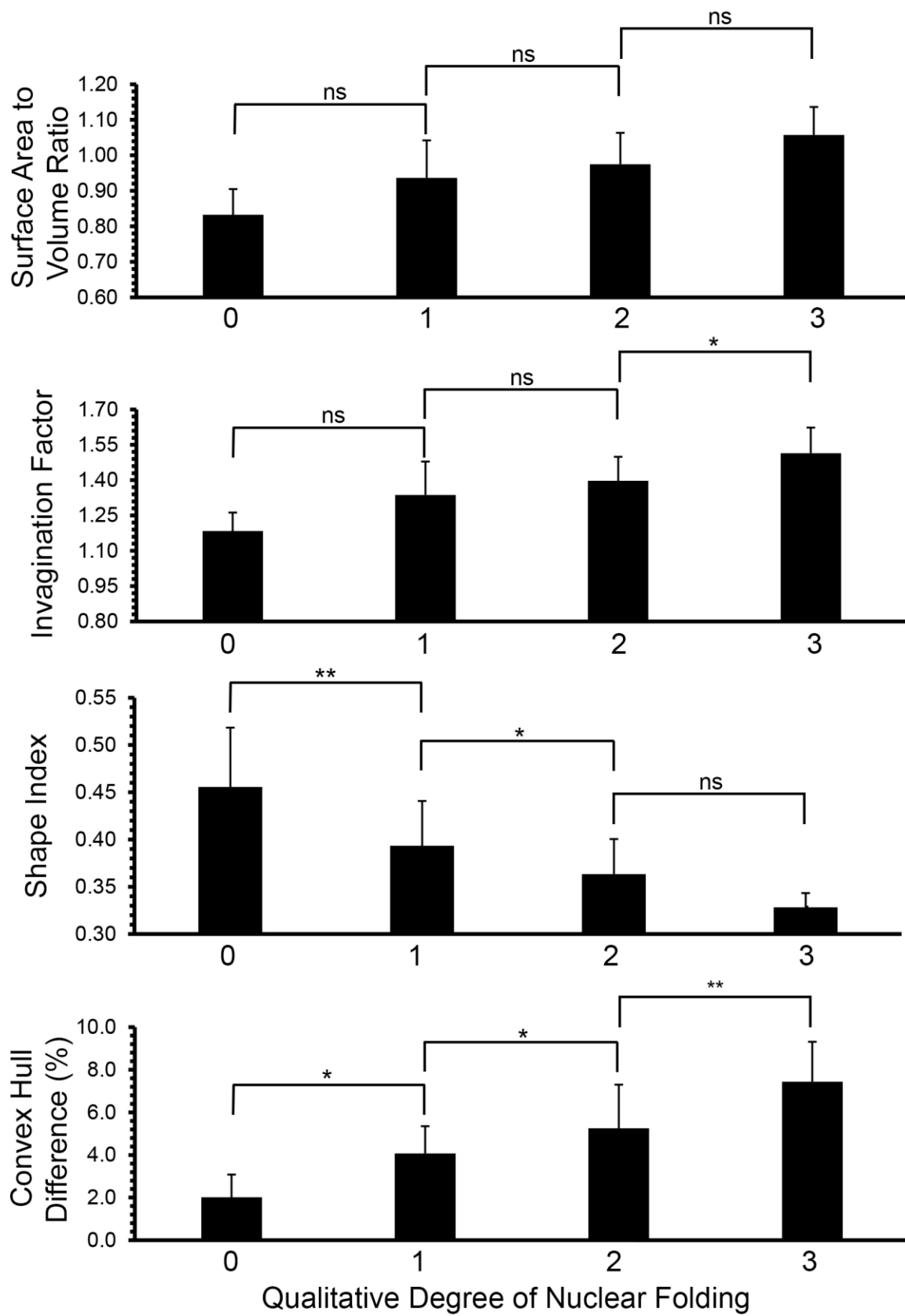
**Figure 3.21. The local shape index scalar field for a heavily invaginated nucleus.**

The local shape index was computed at each triangular vertex of the nuclear mesh using the method described in Chapter 3.2.4.2. Shown here are twelve views from the automatically generated movie of the local shape index scalar field overlaid on the mesh of a heavily invaginated nucleus. The degree of invagination of this nucleus was assigned a score of three on a scale of 0-3 by qualitative observation. Each image represents a successive rotation of  $30^\circ$  about the same axis of the nucleus. The shape index scalar field is coded to a colormap ranging from -1 to +1, as depicted at the bottom of the figure. Negative values indicate that the local shape is more concave, while positive values indicate a more convex local shape. As expected, the local shape index is more negative in the proximity of invaginations.



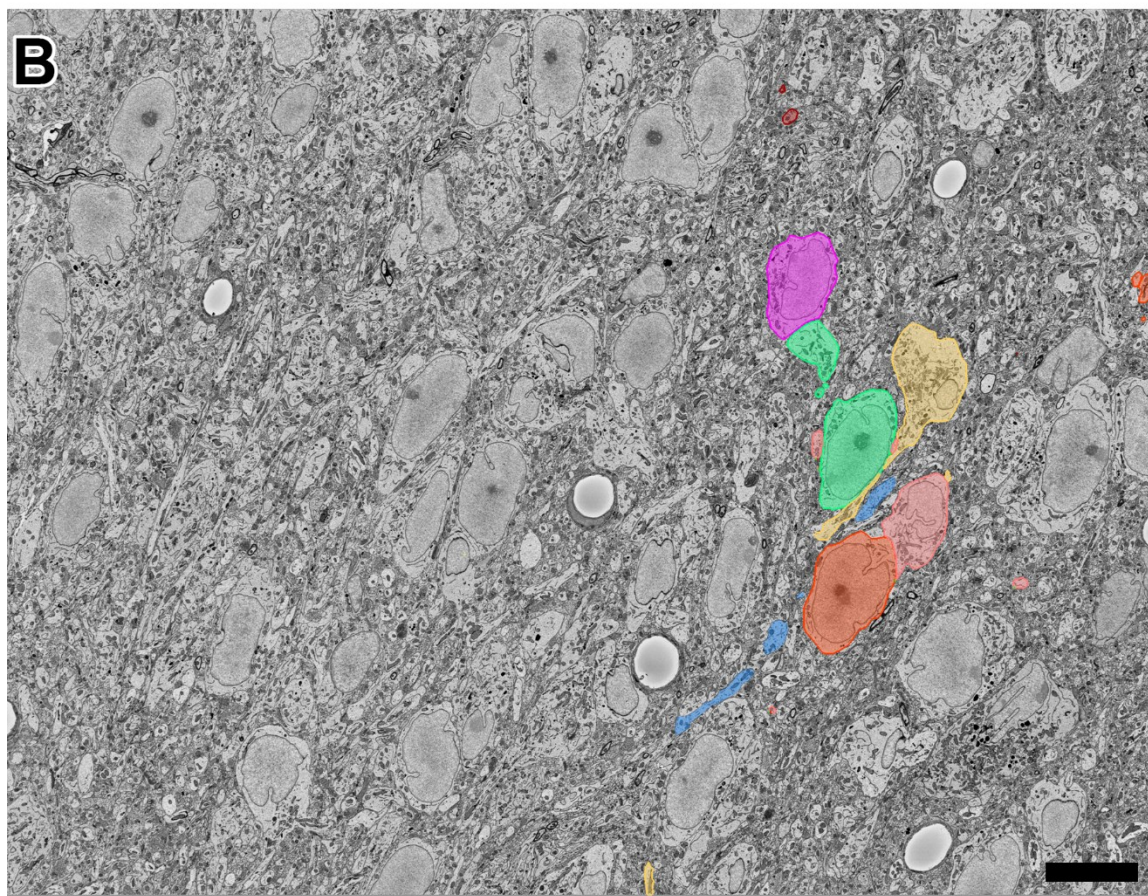
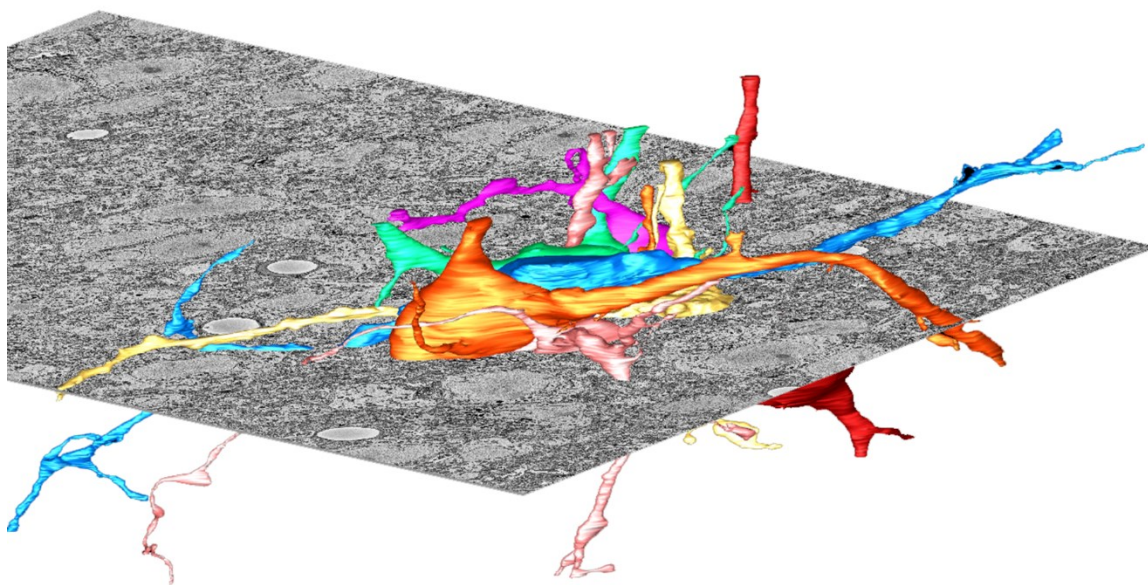


**Figure 3.22. The shape index and convex hull difference are able to discern small qualitative differences in nuclear invagination.** The average SVR,  $IF_{3D}$ , shape index, and CHD are displayed graphically for each qualitatively assessed score of nuclear invagination. Error bars represent the standard deviation. The mean of the SVR was not significantly different between any adjacent score classes. The mean of the  $IF_{3D}$  was not significantly different between score classes 0-1 and 1-2, but was significantly different between score classes 2-3. The opposite was true for the shape index, whose means were significantly different between score classes 0-1 and 1-2, but not significantly different between score classes 2-3. The CHD was the only metric to demonstrate statistically significant differences between all adjacent score classes. (ns: not significant; \*:  $p < 0.05$ ; \*\*:  $p < 0.01$ ).



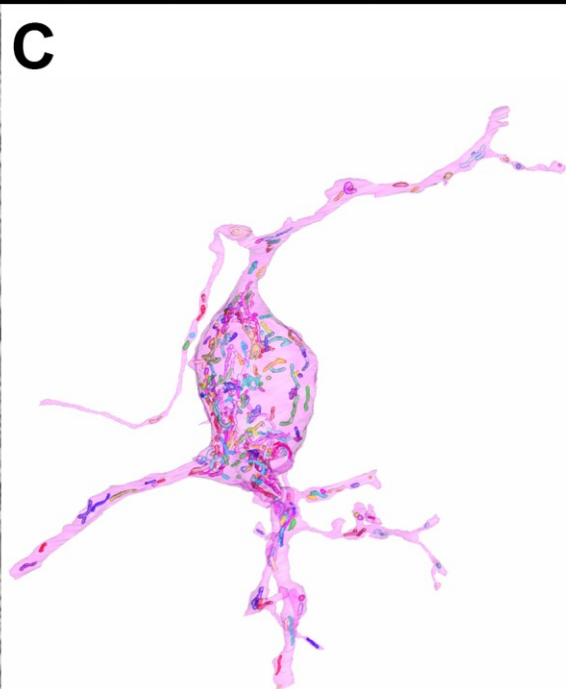
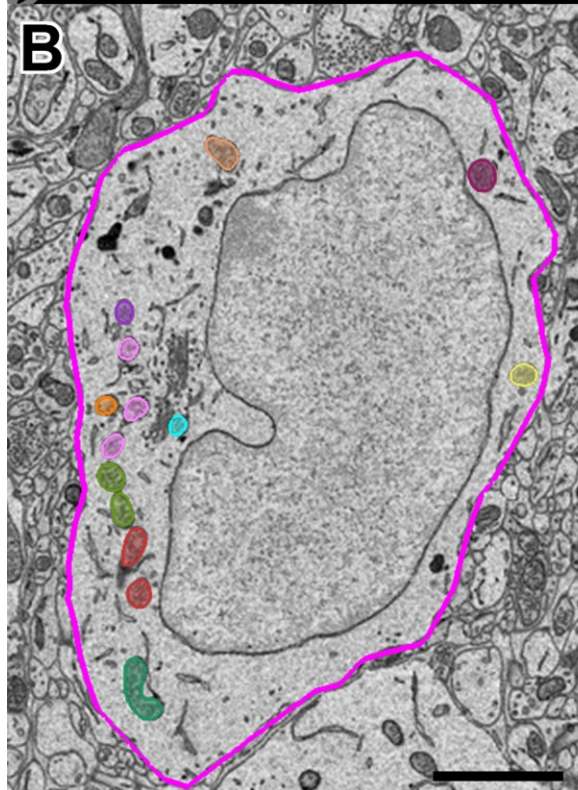
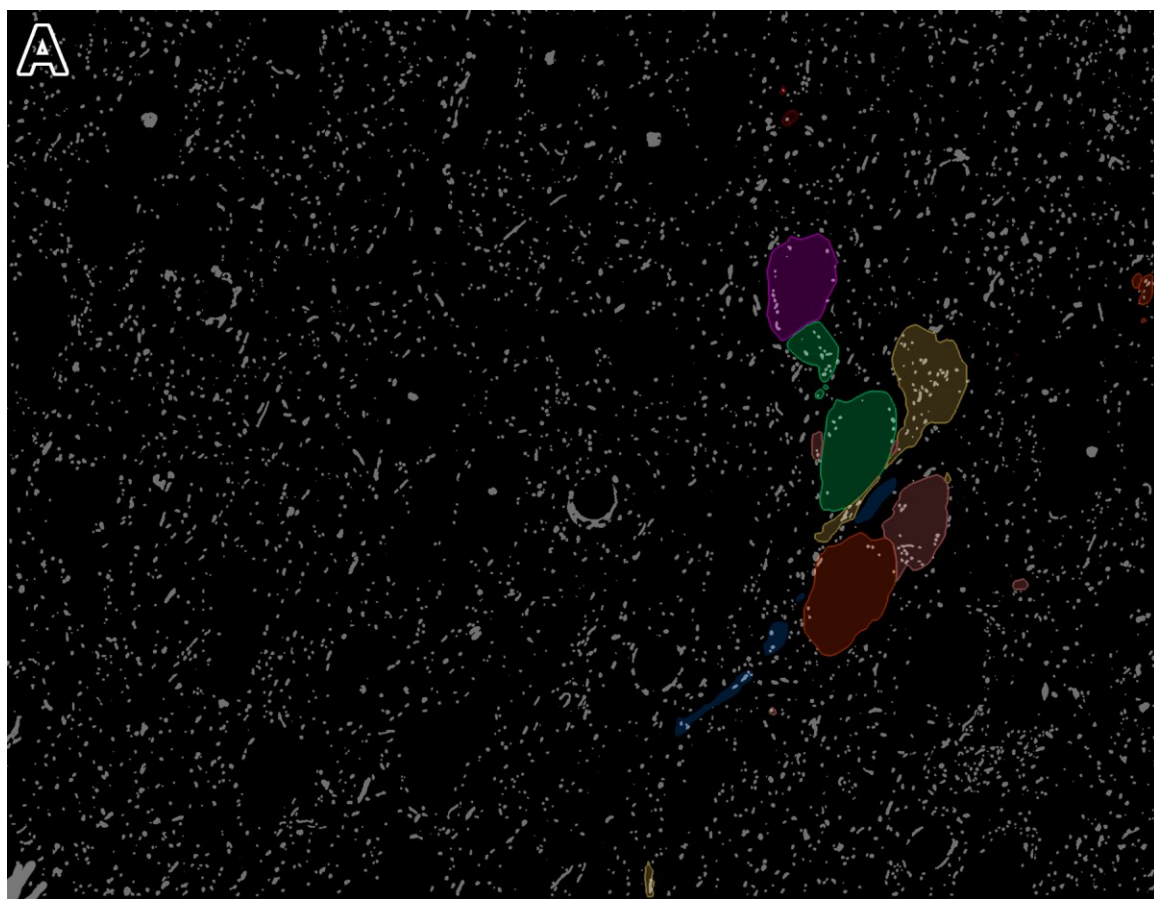
**Figure 3.23. Neuronal compartments were delineated by manual segmentation of the plasmalemma.** Neuronal plasma membranes were manually segmented to provide a means by which automatically segmented organelles could be sorted into their corresponding cells. Contours were manually drawn on sections spaced a variable number of axial steps apart. The number of sections that could be skipped depended upon the feature being traced. When manually tracing a neuronal cell body, as many as 5-10 slices could be skipped without significantly jeopardizing accuracy. However, branched neurites, such as spiny dendrites, often required manually traced contours on every slice or every other slice. Following manual tracing, skipped slices were automatically filled in via an interslice interpolation algorithm. Shown here are surface renderings of seven manually traced neurons from the test SCN dataset (A). These renderings are overlaid on an SBEM slice, and transparent cross-sectional overlays of the renderings are shown on this same slice (B). The overlays in (B) are representative of what would have been manually traced or filled in by interpolation (scale bar = 10  $\mu\text{m}$ ).



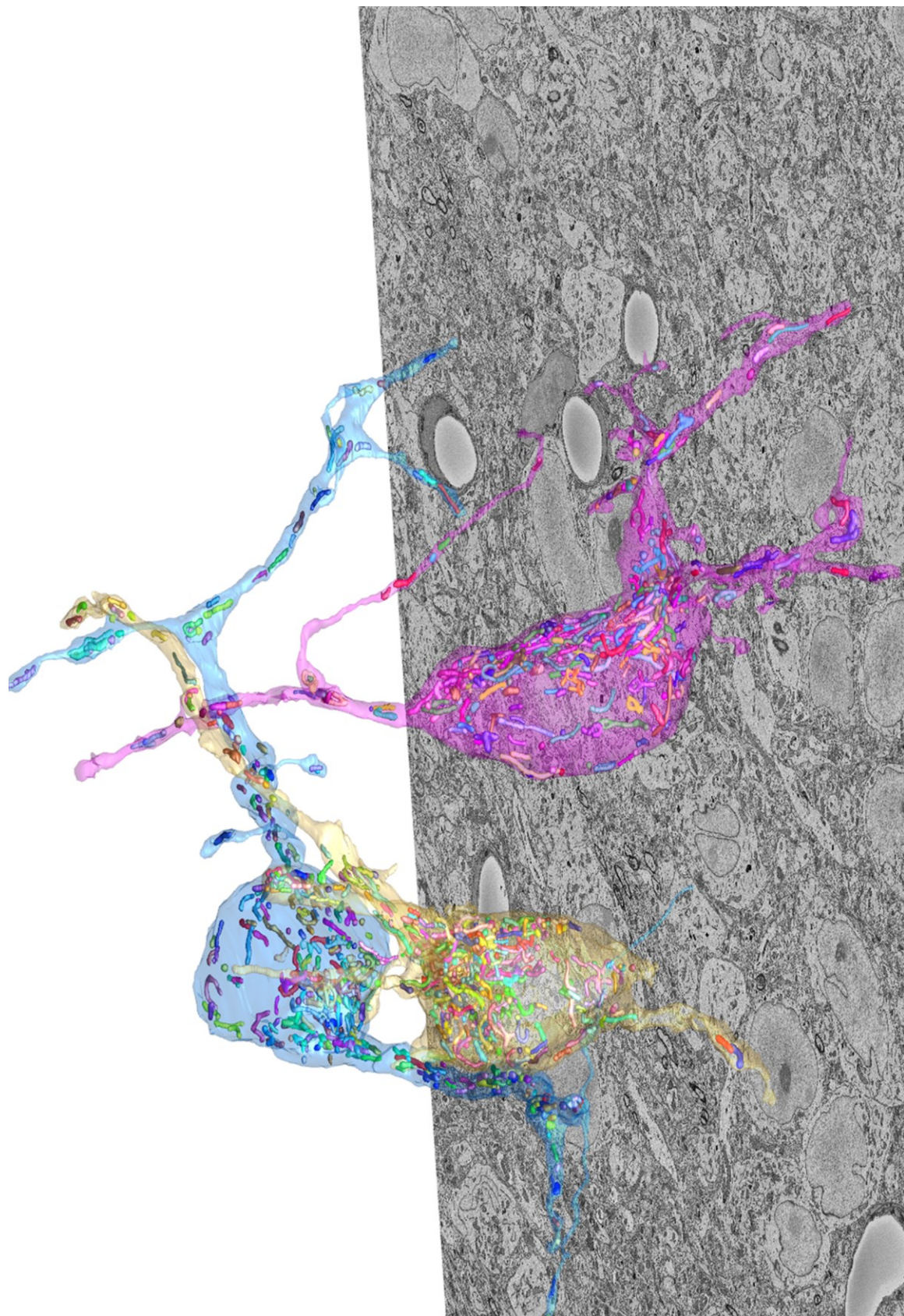
**A**

**Figure 3.24. Binary masks generated from neuronal segmentations are used to separate organelles into their proper cellular compartments.** The segmentations shown in Figure 3.23 were used as binary masks to separate organelles into their proper cells. The intersection of each neuronal binary mask with a single slice of a mitochondrial automatic segmentation (A) demonstrates this process. A cross-section from the same plane of the magenta neuron shown in (A) is overlaid on the original SBEM image in (B) to demonstrate this segregation. The mitochondrial profiles shown here correspond exactly to the binary connected components shown in the mask. All mitochondria falling outside of the mask specified by the magenta neuron are excluded (B). These cell-specific mitochondrial groupings are then meshed, sorted, and manually edited to produce the rendering seen in (C). This rendering consists of all mitochondria contained within the magenta neuron. Each mitochondrial rendering was given a different color to illustrate individual morphologies.





**Figure 3.25. Dataset-wide renderings of mitochondria belonging to three SCN neurons.** The method described in Chapter 3.2.5 was used to generate cell-specific models of mitochondria. Depicted here are three manually segmented neuronal renderings filled with automatically generated mitochondrial reconstructions. The models are overlaid on a slice of the original SBEM image stack. These models can be used to quantify mitochondrial morphology and study how mitochondria are distributed throughout the cell, including in the soma, dendrites, and axon.



### 3.3. Discussion

In the first section of this chapter, the MPAS algorithm for enhancing segmentation performance for isotropic datasets was introduced. Since most SBEM datasets must be downsampled significantly in the lateral dimension to achieve isotropic voxels, this algorithm is most applicable to features such as nuclei and nucleoli, whose segmentation performance does not dramatically decrease with increased pixel sizes (Figure 2.17). For the dataset tested here, the isotropic pixel size was 30 nm, which is too coarse to allow for quality automatic segmentations of mitochondria. It is likely that the improvement provided by the use of the MPAS algorithm would be outweighed by the deleterious effect of image downsampling. For datasets in which the isotropic pixel size is in the range of 15 nm or less, this algorithm should be able to improve segmentation accuracy for all of the organelles tested here. Such datasets can be acquired using FIBSEM or SBEM with a smaller section thickness.

Another drawback of the MPAS method is the increased computational time required, since the automatic segmentation of all images re-sliced about three orthogonal planes requires three times the computational load. This load could potentially be alleviated by performing the re-slicing and orthogonal automatic segmentations in only certain regions of the entire volume. For example, if nuclei are being segmented, one would first run the normal automatic segmentation in the XY plane. If this segmentation is not satisfactory, MPAS could be initiated, but only run on areas of the XY segmentation that clearly possessed nuclei. These regions could be automatically determined by a simple algorithm that looks for regions of maximum probability in the XY probability maps. Such regions would then be subjected to the MPAS algorithm, and the region they were extracted from would be replaced by the averaged MPAS probability map. This implementation should be simple to achieve, and will be explored in the future.



The MSI interslice interpolation algorithm presented here will soon be incorporated into the nuclear automatic segmentation workflow to provide corrections. Poorly segmented slices will be detected by looking for abnormal spikes in the perimeter or cross-sectional area of segmented objects, or by looking for slices in which the centroid shifts dramatically from one section to the next. Alternative metrics, such as descriptors of boundary irregularity, could also prove useful. These sections can then be automatically removed and replaced by interslice interpolations. The whole process of correction can fortunately be parallelized by working on one 3D connected component per processor, a fact that should drastically reduce processing time. This method can also be applied to the correction of segmentations from smaller objects such as mitochondria. However, the rules it would need to follow for the determination of poorly segmented slices would likely be different than those required for larger features such as nuclei.

The rest of the methods presented in this chapter provide a seamless workflow for providing quantifiable models from 2D automatic segmentations. A script for calculating parameters of nuclear morphology and spatial organization was presented, and its results were demonstrated (Appendix D). The average run time for this script is in the range of 5-10 minutes for a full dataset containing hundreds of nuclei, which is trivial when compared to the time required for pixel classification. It also eliminates the need for individual users to learn how to operate multiple analytical software packages, which should make these types of analyses much more accessible to the average scientist. By writing outputs to simple CSV files, the proposed method provides a reproducible bookkeeping system that facilitates easy import into Microsoft Excel and software packages for statistical analyses.

Finally, a workflow for the cell-specific analysis of smaller cytoplasmic organelles, such as mitochondria, was presented. While this method does require segmentations of cell boundaries, such segmentations have the potential to be acquired through some of

the automatic algorithms already present in the literature (Jurrus et al., 2009; Straehle et al., 2011; Andres et al., 2012; Liu et al., 2013). A script similar to the one presented here for nuclei is currently in development for the morphological quantification and spatial characterization of cytoplasmic organelles segmented and separated using this approach. The incorporation of all of the technologies presented here into a seamless, automatic workflow for organelle characterization will be discussed in greater detail in Chapter 5.



## **Chapter 4**

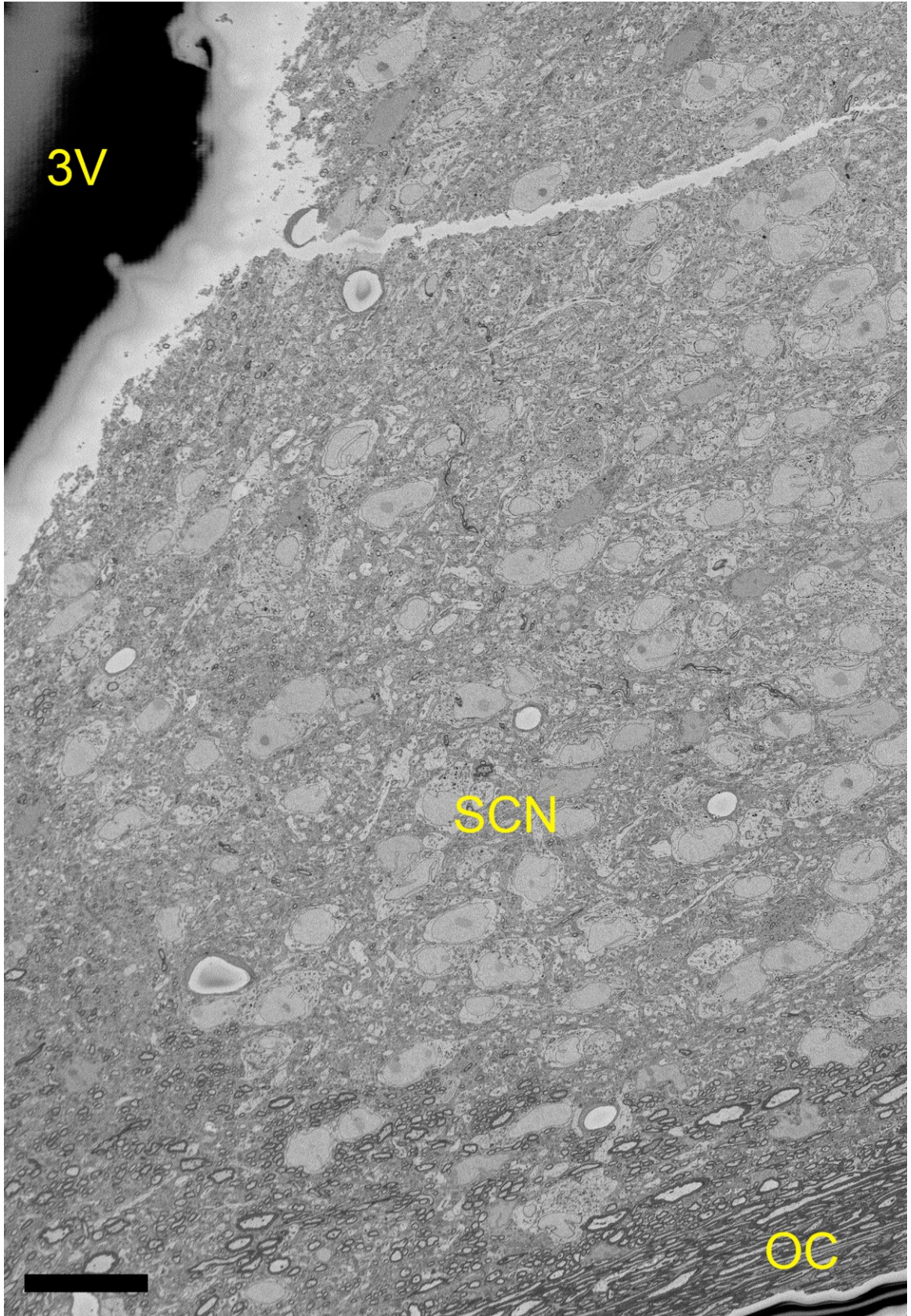
### **Chronomorphological Studies of the Mammalian Suprachiasmatic Nucleus**

## 4.1. Introduction

Living organisms exhibit remarkably accurate rhythms in almost all known biological activities. Such rhythms are the result of an evolutionarily conserved internal, or circadian, clock that can be traced back to earth's most primitive life forms (Mohawk et al., 2012). Circadian control over biological activities has been demonstrated in both unicellular and multicellular organisms, and entire fields are dedicated to the study of rhythms in species as diverse as cyanobacteria, fruit flies, algae, rodents, and humans (Bell-Pedersen, 2005). To be defined as circadian, a biological rhythm must exhibit two important characteristics: (1) it must have a period of roughly 24 hours, and (2) its rhythmicity must persist when removed from environmental influences and placed under constant conditions. This persistent, intrinsic oscillation, known as a free-running rhythm, can be entrained through the detection of environmental cues, or zeitgebers. The most well-known zeitgeber is light, which allows organisms to become entrained to the solar light:dark (LD) cycle. In turn, these entrained rhythms regulate most known physiological parameters, including the sleep-wake cycle, body temperature, athletic ability, feeding behavior, hormone secretion, blood pressure, and glucose metabolism (Takahashi et al., 2008).

Decades of research have definitively identified the suprachiasmatic nucleus (SCN) as the master circadian pacemaker of mammals. The SCN consists of two anterior hypothalamic nuclei of roughly 10,000 neurons each, situated dorsal to the optic chiasm and paired bilaterally across the third ventricle (Figure 4.1). SCN neurons receive light cues via direct innervation from the retinohypothalamic tract, and this information is carried by a specific subset of melanopsin-expressing neurons known as intrinsically photosensitive retinal ganglion cells (ipRGCs) (Hattar et al., 2002; Panda et al., 2005).

**Figure 4.1. An overview of SCN neuroanatomy.** A low magnification image of the block-face collected prior to SBEM image acquisition. The image depicts a single SCN and its surrounding anatomical structures (scale bar = 20  $\mu\text{m}$ ). The SCN sits directly above the optic chiasm (OC) and is paired bilaterally across the third ventricle (3V, other SCN not depicted). The density of cell packing can be used to delineate the SCN from its surrounding hypothalamus; the somata of SCN neurons tend to pack into long clusters, while those of the hypothalamus do not. The SCN is also relatively devoid of myelinated axons, a feature that is especially apparent due to its juxtaposition with the OC.



Ablation of the SCN destroys behavioral rhythmicity in rats (Stephan and Zucker, 1972), and the transplantation of a WT SCN into mutant hamsters with genetically shortened circadian periods restores the mutants to normal periods of roughly 24 hours (Ralph, et al., 1990). Interestingly, the same is true in reverse; transplantation of the SCN from a mutant hamster into a WT animal causes it to adopt the shortened circadian period of the mutant.

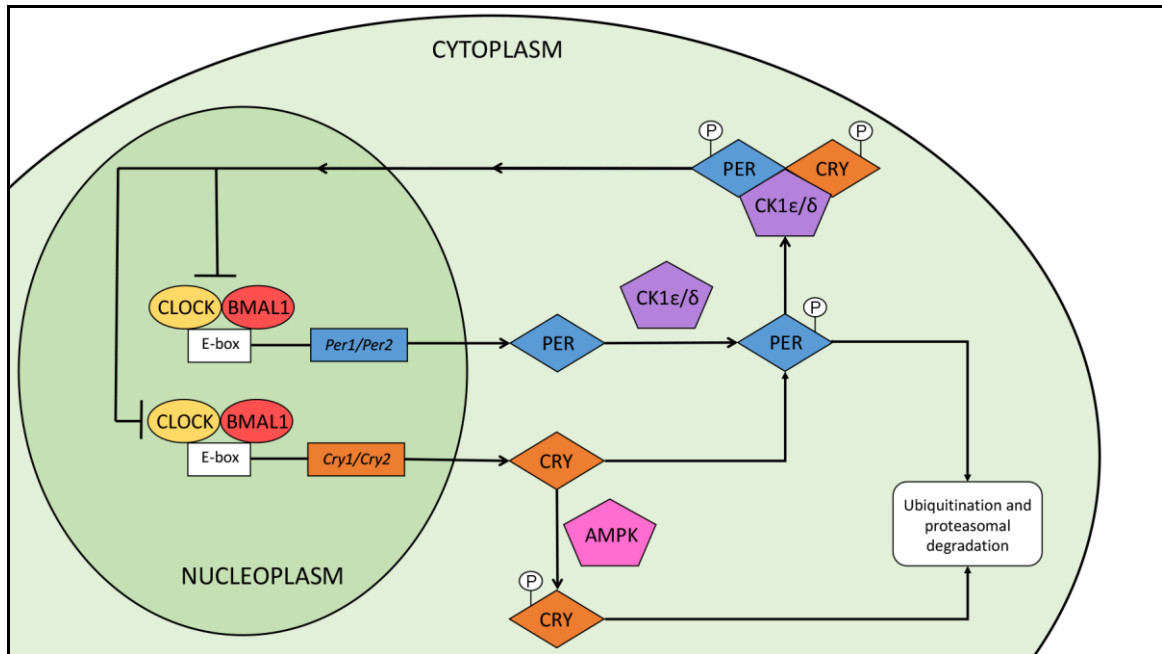
Although these studies provided intriguing evidence that the period of mammalian circadian rhythms is determined by the SCN, the question of how the SCN synchronizes the rest of the body to operate under the same clock remained. When neurons from SCN explants are dissociated and grown in culture, they demonstrate independently phased firing rhythms and rhythms of gene expression that can persist without dampening for up to 50 days (Welsh et al., 1995; Yamazaki et al., 2002). However, the same did not appear to hold true for explants from other tissues; circadian gene expression in cultured fibroblasts was inexplicably found to be detectable only following a change of the culture medium, and this rhythm disappeared within a few days (Balsalobre et al, 1998). However, it was later proven that this lack of rhythmicity was simply an artifact of looking at the gene expression data of the collection of cells as a whole. A subsequent study employing single-cell resolution demonstrated that individual fibroblasts did indeed show rhythmicity, but that these rhythms were out of phase with one another (Welsh et al., 2004). Thus, when looking at the aggregate expression data of all cells, the summation of out-of-phase oscillations appeared arrhythmic. Therefore, the oscillation that Balsalobre and colleagues saw following the change of culture medium was due to synchronization of the independent phases of the fibroblasts, and not to the generation of new rhythms within individual cells. Subsequent studies determined that, like these fibroblasts, nearly all mammalian cells possess their own cell-autonomous circadian clocks, and that it is the

SCN's job to synchronize these so-called peripheral clocks to the light cues it receives from the retina. This is achieved in part due to the phase diversity of firing rates among SCN neurons, which allows the SCN to provide differently phased outputs to different tissues and organ systems (Welsh et al., 2010).

At the molecular level, the mechanism of the mammalian circadian clock is governed by a negative-feedback transcriptional loop consisting of proteins whose expressions oscillate rhythmically. At the beginning of the cycle, the transcriptional activators CLOCK and BMAL1 interact with one another to up-regulate expression of transcription factors belonging to the Period (PER1, PER2) and Cryptochrome families (CRY1, CRY2). These *Per* and *Cry* gene products undergo nucleocytoplasmic export and are translated in the cytosol, where their protein products accumulate as the cycle progresses and dimerize with one another. PER-CRY heterodimers are subsequently imported back into the nucleus, where they act to repress the CLOCK/BMAL1 complex and suppress their own transcription (Mohawk et al., 2012). PER and CRY are progressively phosphorylated as the cycle progresses, and such phosphorylation events target them for ubiquitination and degradation by proteasomes (Takahashi et al., 2008). Once this occurs, CLOCK and BMAL1 renew their upregulation of PER and CRY expression, restarting the feedback loop. This whole cycle, of course, has a period of roughly 24 hours, and is known to be present in both SCN neurons and cells of peripheral tissues. Mutations to components of this loop cause significant mammalian arrhythmias; the double knockout of both the *Cry1* and *Cry2* genes, for example, abolishes free-running rhythms in mice (van der Horst et al., 1999).

Components of this core feedback loop, however, aren't the only mammalian proteins whose expressions display significant circadian rhythmicity. In 2002, Panda and





**Figure 4.2. The core transcriptional-translational feedback loop of the mammalian circadian clock.** Mammalian cells possess a delicately balanced negative-feedback loop consisting of rhythmically expressed proteins. Inside the nucleus, the transcriptional activators CLOCK and BMAL1 form a complex with one another, and this complex acts to up-regulate the expression of the PER and CRY transcription factors by binding to the E-box of their genes. *Per* and *Cry* transcripts then undergo nucleocytoplasmic export and are translated in the cytosol. While in the cytosol, PER undergoes phosphorylation by casein kinase 1 (CK1)  $\epsilon$  and CK1 $\delta$ . Phosphorylated PER can undergo heterodimerization with CRY, and this PER/CRY complex is able to achieve nucleocytoplasmic import. Within the nucleoplasm, the PER/CRY complex interacts with the CLOCK/BMAL1 complex in a way that inhibits the subsequent transcription of PER and CRY. As time passes and the PER/CRY complex degrades in the nucleoplasm, this inhibition is removed and the cycle is able to begin anew.

colleagues used gene expression profiling to demonstrate the rhythmic expression of over 650 gene transcripts in the mouse SCN and liver (Panda et al., 2012). Interestingly, most cycling transcripts were specific to either the SCN or the liver; only 28 transcripts cycled in both tissues. Furthermore, the cycling transcripts that are unique to either the SCN or liver play critical roles in the physiological function of their tissue. Examples of cycling transcripts in the SCN are those that control neuropeptide synthesis, while genes involved in sugar metabolism cycle in the liver. This demonstrates the ability of peripheral clocks, such as those of the liver, to oscillate in a manner that suits their own physiological demands.

Since this study and a number of others since it (Storch et al., 2002; Miller et al., 2007; Hughes et al., 2009) have established that the levels of as many as 3-10% of all mRNAs in a given tissue are governed by the circadian clock (Mohawk et al., 2012), it is reasonable to ask if such oscillations are also present at the level of tissue ultrastructure. From the 1970s to the 1980s, long before such molecular-level details were known, a number of researchers intrigued with answering this very question established the field of chromorphology – the study of how biological structures change with respect to the circadian cycle. In his review of the field in 1983, the German chromorphologist Heinz von Mayersbach wrote the following:

Temporal variations in such biological components as hormones and enzymes, for example, are undoubtedly the expression of temporal alterations in metabolic processes. Since metabolic processes are based on cellular functions, the question arises: To what extent are structural manifestations during the circadian cycle visible at the cellular level? (von Mayersbach, 1983)

Despite the fact that it was posed over 30 years ago, this question remains largely unanswered, and the study of chromorphology at the ultrastructural level has mostly fallen by the wayside.



Though recent evidence supports the day-night plasticity of SCN neurons at the level of their glial coverage (Becquet et al., 2008; Girardet et al., 2014), no known work investigating plasticity at the level of subcellular ultrastructure has been performed. A possible explanation for this is that, prior to the introduction of block-face imaging, the acquisition of 3D EM datasets comprising enough cells to achieve statistical relevance was a monumental task. Early chromomorphological studies relied on 2D stereology to compute the volume fractions of subcellular components; 3D morphologies could only be estimated by making geometrical assumptions, such as the approximation of the nucleus as a sphere or ellipsoid.

This chapter will describe the application of the technologies developed in this dissertation to the study of SCN neuronal chromomorphology. Organelle morphologies were quantified and compared using SBEM datasets, and electron tomography was employed as a complementary technique. The rest of this introduction will contain a brief overview of SCN neuroanatomy followed by a survey of findings from previous chromomorphological studies of the brain and peripheral organ systems.

#### **4.1.1. Neuroanatomy of the suprachiasmatic nucleus**

The first comprehensive studies of SCN neuroanatomy were conducted by Fritz Güldner (Güldner, 1976) and Anthony van den Pol (van den Pol, 1980), both of whom used the rat as a model organism. Through the use of a variety of techniques, including Golgi impregnation, Nissl stains, and high resolution EM, van den Pol's work provided important insight into not only the gross anatomy of the rat SCN, but also the structure and microanatomy of its neurons and glial cells. Though its delineation from the surrounding hypothalamus is difficult to the untrained eye, van den Pol presented a number of unique anatomical features that can be used to identify the SCN. First, SCN

neurons exhibit tight packing and frequently form chains in which multiple somata are in apposition with one another or separated only by thin glial processes; this cell packing is even more prominent in the dorsomedial (DM) regions of the nucleus (van den Pol, 1980). While an obvious “cell-free” zone devoid of such cell bodies separates the third ventricle from the SCN, the other boundaries are often less obviously demarcated and must typically be identified based on changes in cell body apposition. A final telling sign is that the SCN contains significantly fewer myelinated axons than the surrounding hypothalamus.

The dendrites and axons of most SCN neurons terminate locally within the nucleus, though dendrites extending into the hypothalamus are found in its dorsal and lateral regions (van den Pol, 1980). Axons are derived from somata and dendrites in a roughly 50/50 ratio and create both terminal and *en passant* boutons that may spread diffusely throughout the nucleus or establish a more restricted field of influence near their soma. Through his observations of Golgi impregnated tissue, van den Pol grouped SCN neurons into five classes, distinguished primarily by the morphologies of their neurites:

- (1) Simple bipolar cells, which possess two primary dendrites at opposite sides of the soma. They are mostly devoid of spines and have dendrites that rarely branch.
- (2) Monopolar cells, which have a single primary dendrite that bifurcates into many smaller, distal dendrites
- (3) Curly bipolar cells, whose dendrites often bend and change directions. Spines are typically present on both dendrites and soma.
- (4) Radial multipolar cells, which possess many dendrites that radially extend from the soma.

- (5) Spiny neurons, which are the most amorphous of the classes and possess predominantly spherical cell bodies with spines and appendages of varying shapes and sizes.

The majority of SCN neurons have multiple nucleoli, which are situated on opposing sides of the nucleus and in close proximity to the nuclear envelope. In cells with a single nucleolus, the nucleolus is situated more towards the middle of the nucleus. SCN neurons and astrocytes demonstrate significantly invaginated nuclear membranes, a morphological trait that serves to increase the surface area-to-volume (SAV) ratio as well as decrease the distance from nucleoli to the nuclear periphery. (van den Pol, 1980).

Traditionally, each ellipsoidal SCN is split into two anatomical subdivisions: (1) a ventrolateral (VL) shell, and (2) a dorsomedial (DM) core (Moore et al., 2002; Colwell, 2011). Though it is believed that all SCN neurons synthesize GABA (Morin, 2013), their synthesis of other neuropeptides has historically formed the basis for the demarcation of these subdivisions. Neurons in the core, which lies adjacent to the optic chiasm, generally produce vasoactive intestinal polypeptide (VIP) and gastrin-releasing peptide (GRP). This central core is surrounded by a shell of neurons that predominantly produce arginine vasopressin (AVP). These core/shell delineations were first made by employing immunocytochemistry with antibodies specific to the aforementioned neuropeptides (Moore et al., 2002). However, recent groups have argued that this classical model is inaccurate, and a more accurate topographical classification schema is needed (Morin, 2007; Hundahl et al., 2010). For example, neuronal phenotypes and patterns of rhythmicity do not directly correlate with neuropeptide chemistry. Additionally, such anatomical patterns of neuropeptide chemistry are not identical across organisms, differing even amongst species of rodents (Morin, 2013).

Güldner first estimated the volume of a single SCN to be  $0.05 \text{ mm}^3$ ; a value he arrived at by measuring the major and minor axes of the SCN and approximating it as an ellipsoid (Güldner, 1976). This number was refined by van den Pol, who, using camera lucida tracings from serial paraffin sections, arrived at a value of  $0.068 \text{ mm}^3$ . Subsequent measurements have fallen within the range of  $0.02 - 0.07 \text{ mm}^3$ , with variations likely due to differing methods for specimen preparation (Madeira et al., 1995). The rat retina, a tissue that has received the bulk of attention from the connectomics community, has an estimated volume of roughly  $16 \text{ mm}^3$  (Mayhew and Astle, 1997). Thus, when compared to the retina, the SCN presents a size that is significantly more amenable to large-scale reconstructions.

#### **4.1.2. A history of biological chronomorphology**

A circadian influence on cell cycle progression has been demonstrated for a variety of tissue types in numerous organisms, including plants, cyanobacteria, zebrafish, and rodents (Masri et al., 2013). Mitotic activity, as measured by the number of mitotic events per 1,000 cells, is rhythmically controlled by tissue-specific zeitgebers. For example, the mitotic index in the corneal epithelium of rats closely follows the light cycle, while the same index closely follows the feeding pattern in the liver (Philippens, 1980). Given the functions of these two systems, and that mRNA transcripts are known to cycle in a tissue-dependent manner (Panda et al., 2002), such findings are intuitive. Importantly, since progression through the cell cycle involves significant changes in nuclear architecture, it stands to reason that nuclear morphologies are also under a degree of circadian control in non-senescent cell types. Indeed, reports demonstrating pronounced rhythmic variations of nuclear size in liver hepatocytes had surfaced as early as the 1930s

(von Mayersbach, 1983). Rhythms in the percentage of hepatocytes with a binuclear phenotype have been reported in rats, with a binucleation peak occurring during the subjective night and a trough occurring during the subjective day (Röver and Philippens, 1979). The authors attempted to explain these data by hypothesizing an oscillating sequence of amitotic nuclear divisions and nuclear fusion processes. At a molecular level, the extent to which  $^{32}\text{P}$ -labeled precursors (Barnum et al., 1958) and tritiated thymidine (Eling, 1967) are incorporated into the DNA of rat hepatocytes depends significantly upon the time of injection.

These early findings sparked a number of subsequent studies exploring alterations in nuclear morphology across the diurnal cycle in a variety of tissues. Diehl reported daily fluctuations in the nuclear volume of rat pinealocytes, and that pinealocytes in the medulla and cortex demonstrate different patterns of fluctuation (Diehl, 1981). Ensuing studies were in disagreement as to the phasing of this pattern, with peaks of pinealocyte nuclear volume reportedly occurring at the transition between the light and dark phases (Diehl et al., 1984), during the middle of the light phase (Lew et al., 1984; Hira et al., 1989), and during the dark phase (Karasek et al., 1990). One possible explanation for these discrepancies is that each study used slightly different specimen preparation and quantification methods. Another is that different LD cycles were used; Karasek and colleagues used a 14 hr:12 hr LD cycle, while all other studies used the more standard 12 hr:12 hr LD cycle. Bimodal fluctuations in nuclear morphology were reported in rat thyroid follicular cells (Murakami and Uchiyama, 1986), juxtaglomerular cells of the kidney (Watanabe et al., 1988), pancreatic islet cells (Watanabe and Uchiyama, 1988), and alveolar type II cells (Ishii et al., 1989). Importantly, these findings demonstrate the need to sample with a high temporal resolution; if only 2-4 time points are sampled, bimodal behaviors would likely be masked or incorrectly classified as unimodal.

Interestingly, Lew and coworkers reported a qualitative finding that pinealocyte nuclei were more heavily invaginated during the middle of the light period, a morphological factor that had not previously been considered oscillatory (Lew et al., 1984). A study of tadpole nuclei showed that thyroid follicular cells oscillate between small and elongated to large and spherical during a 12 hr:12 hr LD cycle (Wright et al., 1995). The elongated morphology was observed one hour after the onset of light, and the spherical morphology was observed during the dark period and late light period. Nuclei demonstrating the spherical morphology also stained lighter with toluidine blue and had more visible chromatin. These findings suggest that daily fluctuations in nuclear morphology may involve concomitant changes in sub-nuclear compartmentalization, affecting parameters such as chromosome positioning and the location of chromosome territories (Cremer and Cremer, 2001).

Though most of the aforementioned studies arrived at their quantifications using thin section EM combined with stereological methods, a few used traced measurements of nuclei combined with geometrical approximations. Kirillov and Kurilenko approximated nuclear volume in cells of the adrenal cortex using the formula for a rotary ellipsoid:

$$V_{\text{nuc}} = \frac{\pi}{6} L^2 B$$

in which L is the maximum diameter and B is the minimum diameter of the nucleus in cross-section, as determined by measurements from hematoxylin and eosin (H&E) stained, paraffin-embedded sections (Kirillov and Kurilenko, 1977; Kirillov and Kurilenko, 1979). Diehl subsequently employed the same formula in his study of nuclear volume in rat pinealocytes (Diehl, 1981; Diehl et al., 1984). Becker and Vollrath, operating with tissue from the same staining method, employed a formula for nuclear volume that accounts for the circumference, as determined by cross-sectional tracings:

$$V_{\text{nuc}} = \frac{4}{9\pi} (C + \sqrt{\pi \cdot A}) \cdot A$$

Here, K is a constant and C and A denote the circumference and area of the 2D nuclear profile, respectively (Becker and Vollrath, 1983).

Though none have been as extensively studied as the nucleus, a plethora of chromomorphological studies have been conducted on other organelles. In studies conducted on rat lung (Ishii et al., 1989) and porcine pineal gland (Lewczuk et al., 2004) tissue, mitochondrial volume fractions were found to fluctuate and peak during the time of activity of the animal. Lysosomes have been shown to exhibit circadian variations in number, size, activity, and position within hepatocytes of the rat liver lobule, with a peak in number that occurs at the end of the light period (Groh and von Mayersbach 1981). Since autophagic processes also peak during the light period in the liver of nocturnal rodents, the phasing of this maximum in lysosomal quantity makes intuitive sense (von Mayersbach, 1983). Armstrong and Hatton reported that the percentage of cells with multiple nucleoli peaks during the dark phase in the rat supraoptic nucleus (Armstrong and Hatton, 1978). However, this study was likely confounded by the use of only single sections for quantification. Raymond Seïte and his colleagues published a series of studies demonstrating nucleolar volume fluctuations in a variety of rat tissues (Pébusque and Seïte, 1981a; Pébusque and Seïte, 1981b; Bessone and Seïte, 1985; Pébusque and Seïte, 1985; Robaglia and Seïte, 1985). Significant and differentially phased rhythms of nucleolar volume were found in all tissues studied except the nodose ganglion (Pébusque and Seïte, 1985). A comprehensive listing of other significant chromomorphological studies and their relevant findings is given in Table 4.1.

**Table 4.1. A survey of significant chromomorphological studies and their pertinent results.** The table of published studies is sorted first in alphabetical order by the organelle studied, and then in chronological order within each unique organelle. A time frame ranging from 1959 to 2010 is covered.

Citation	Organelle	Parameter Measured	Organism	Anatomical Region	Finding Reported
Murakami and Uchiyama, 1986	Golgi apparatus	Volume	Rat	Thyroid	Bimodal distribution, with peaks during both the light and dark periods.
Lewczuk et al., 2004	Golgi apparatus	Volume	Pig	Pineal gland	Relative volume is higher during the light cycle.
Gerasimov et al., 2010	Golgi apparatus	Volume	Rat	Pineal gland	Volume decreases after 45 days of exposure to constant light, and normalizes to control levels after 90 days of exposure to constant light.
Müller and Gerber, 1985	Golgi vesicles	Number per cell	Rat	Pancreas	Total number increase steeply between ZT18 and ZT22, and then declines. No rhythm in cisternal Golgi was detected.
Benson and Krasovich, 1977	Granulated vesicles	Number per cell	Mouse	Pineal gland	A three- to four-fold increase in the number of granulated vesicles in pinealocytes during the late light period.
Groh and von Mayersbach, 1981	Lysosomes	Number per cell	Rat	Liver	Changes in number and size of liver lysosomes by immunohistochemistry. Peak in number at the end of the resting period.
Karasek et al., 1990	Lysosomes	Cross-sectional area	Rat	Pineal gland	No significant change detected between day and night.



**Table 4.1. A survey of significant chromomorphological studies and their pertinent results, Continued.**

Citation	Organelle	Parameter Measured	Organism	Anatomical Region	Finding Reported
Ishii et al., 1989	Mitochondria	Volume	Rat	Lung	Volume density fluctuates over 24-hours, with a maximum in the dark cycle.
Karasek et al., 1990	Mitochondria	Cross-sectional area	Rat	Pineal gland	Peak at nighttime in animals on a 14h:10h LD cycle.
Lewczuk et al., 2004	Mitochondria	Volume	Pig	Pineal gland	Relative volume is significantly highest six hours into the light period.
Gerasimov et al., 2010	Mitochondria	Volume	Rat	Pineal gland	Total volume decreases following both 45 and 90 days of constant light.
Armstrong and Hatton, 1978	Nucleolus	Number per cell	Rat	Anterior supraoptic nucleus	Percentage of cells with multiple nucleoli peaks at ZT18, corresponding to a peak in plasma corticosterone levels.
Pébusque and Seïte, 1981a	Nucleolus	Volume	Rat	Superior cervical ganglion	Diurnal rhythm of mean nucleolar volume, with a peak roughly halfway through the subjective night.
Pébusque and Seïte, 1981b	Nucleolus	Volume	Rat	Superior cervical ganglion	Expanded upon the previous study, reporting rhythms in the volume of fibrillar centers and the granular and vacuolar components of the nucleolus.
Lew et al., 1982	Nucleolus	Volume	Rat	Pineal gland	Peaks in total nucleolar volume and granular component volume occurred at the onset of light

**Table 4.1. A survey of significant chromomorphological studies and their pertinent results, Continued.**

Citation	Organelle	Parameter Measured	Organism	Anatomical Region	Finding Reported
Bessone and Seite, 1985	Nucleolus	Volume	Rat	Supraoptic nucleus	Volume peaks in the middle of the dark cycle and troughs at the beginning of the dark cycle.
Pébusque and Seite, 1985	Nucleolus	Volume	Rat	Nodose ganglion	No rhythm in neuronal nucleolar volume was detected.
Robaglia and Seite, 1985	Nucleolus	Volume	Rat	Adrenal medulla	A hypothesized ultradian rhythm for nucleolar volume in chromaffin cells.
Karasek et al., 1990	Nucleolus	Cross-sectional area	Rat	Pineal gland	Peak at nighttime in animals on a 14h:10h LD cycle.
Hellman and Hellerström, 1959	Nucleus	Cross-sectional area	Rat	Pancreas	Nuclear area trends towards higher values in the later hours of daylight in islet A-cells.
Hardeiland et al., 1973	Nucleus	Cross-sectional area	Rat	Liver	Nuclear area peaks during the light cycle, roughly four hours after the onset of light.
Kirillov and Kurilenko, 1977	Nucleus	Volume	Mouse	Adrenal cortex	The nuclear volume increases at night and decreases during daytime in the zona fasciculata.
Kirillov and Kurilenko, 1979	Nucleus	Volume	Mouse	Adrenal cortex	Injections of adrenocorticotrophic hormone (ACTH) enlarge nuclei in the zona fasciculata and disrupt their volumetric rhythmicity.

**Table 4.1. A survey of significant chromomorphological studies and their pertinent results, Continued.**

Citation	Organelle	Parameter Measured	Organism	Anatomical Region	Finding Reported
Diehl, 1981	Nucleus	Volume	Rat	Pineal gland	Fluctuations of nuclear size in pinealocytes throughout the day. Central and peripheral cells exhibit different patterns.
Becker and Vollrath, 1983	Nucleus	Volume	Rat	Pineal gland	Rhythms in nuclear volume are bimodal for peripheral pinealocytes. In central cells, the rhythm either matched that of the peripheral cells or was non-existent.
Diehl et al., 1984	Nucleus	Volume	Rat	Pineal gland	Increase in nuclear volume at the transition from light to dark, and a decrease at the transition from dark to light. Changes were tracked over 10 days.
Lew et al., 1984	Nucleus	Surface area	Rat	Pineal gland	Peak in cross-sectional surface area of pinealocyte nuclei during the middle of the light period. Nuclear membrane folding was more pronounced during this time.
Murakami and Uchiyama, 1986	Nucleus	Volume	Rat	Thyroid	Bimodal distribution, with peaks two hours before and six hours after light onset.
Watanabe et al., 1988	Nucleus	Volume	Rat	Kidney	Volume of juxtaglomerular cell nuclei has a bimodal pattern, with peaks in both the light and dark cycles
Watanabe and Uchiyama, 1988	Nucleus	Diameter	Rat	Pancreas	Diameters of islet A- and B-cells are maximal at the end of the light period and middle of the dark period, respectively.

**Table 4.1. A survey of significant chromomorphological studies and their pertinent results, Continued.**

Citation	Organelle	Parameter Measured	Organism	Anatomical Region	Finding Reported
Hira et al., 1989	Nucleus	Cross-sectional area	Hamster	Pineal gland	Peak in the area of nuclear profiles during the middle of light period in young animals. No rhythms detected in old animals (120-130 days old).
Ishii et al., 1989	Nucleus	Diameter	Rat	Lung	Nuclear diameter exhibits a bimodal pattern over 24 hours in alveolar type II cells.
Dolci et al., 1990	Nucleus	Volume fraction	Rat	Exocrine pancreas	No significant diurnal rhythm detected.
Karasek et al., 1990	Nucleus	Cross-sectional area	Rat	Pineal gland	Peak at nighttime in animals on a 14h:10h LD cycle.
Wright et al., 1995	Nucleus	Morphology	Tadpole	Thyroid	The authors reported qualitative morphological differences in nuclei at different time points, changing from small and elongated to large and round.
Gerasimov et al., 2010	Nucleus	Volume	Rat	Pineal gland	Volume decreases after 45 days of exposure to constant light.
Müller and Gerber, 1985	Ribosomes	Number per cell	Rat	Pancreas	Unimodal rhythm, with a peak at ZT18, corresponding to the first peak in rough ER volume.

Table 4.1. A survey of significant chromomorphological studies and their pertinent results, Continued.

Citation	Organelle	Parameter Measured	Organism	Anatomical Region	Finding Reported
Müller and Gerber, 1985	Rough ER	Volume	Rat	Pancreas	Bimodal rhythm of RER volume, with peaks at ZT18 and ZT6.
Murakami and Uchiyama, 1986	Rough ER	Volume	Rat	Thyroid	Both volume and surface densities vary, with peaks occurring during the dark and light cycles, respectively.
Watanabe et al., 1988	Rough ER	Surface density	Rat	Kidney	Peak occurs two hours into the dark cycle in juxtaglomerular cells. This corresponds with a peak in nuclear volume.
Ishii et al., 1989	Rough ER	Volume	Rat	Lung	Volume and surface densities peak during the dark period and trough during the light period in alveolar type II cells.
Karasek et al., 1990	Rough ER	Cross-sectional area	Rat	Pineal gland	Peak at nighttime in animals on a 14h:10h LD cycle. Greater than two-fold increase.
Dolci et al., 1990	Zymogen granule	Volume fraction	Rat	Exocrine pancreas	Significant circadian periodicity, with a peak at the end of the light period.

## 4.2. Methods

### 4.2.1. Tissue processing and SBEM imaging

Wildtype C57BL/6J mice were housed in light-impermeable boxes and entrained to a 12:12-hr light-dark (LD) cycle as previously described (Miller et al., 2007). The local time of “lights on” was 6:00 A.M. (ZT0), while the corresponding time of “lights off” was 6:00 P.M. (ZT12). Following entrainment, mice were isolated in a manner that did not expose them to ambient white light, anesthetized, and transcardially perfused at two distinct time points in the diurnal cycle: ZT4-6 and ZT16-18. Time ranges are used in reporting due to the uncertain delay between animal isolation and perfusion. The VL SCN of two animals at each time point were harvested and prepared using a standard protocol (Wilke et al., 2013). The resin-embedded tissue was mounted on an aluminum specimen pin and prepared for SBEM imaging as previously described (Holcomb, et al., 2013). Animal perfusions, dissections, and tissue preparation steps were performed by Eric Bushong and Keun-Young Kim at NCMIR.

The first ZT4-6 SBEM dataset and both ZT16-18 datasets were collected using a raster size of 32,000 x 24,000 pixels, an axial step size of 30 nm, a magnification of 800x, a lateral pixel size of 3.899 nm/pixel, a spot size of 1.0, and a pixel dwell time of 500 ns (ZT4-6 animal #1: CCDBID 81739; ZT16-18 animal #1: CCDBID 90850; ZT16-18 animal #2: CCDBID 93678). The corresponding accelerating voltage used and number of sections collected are reported in Table 4.2. Low magnification images of the block-face were collected before and after SBEM imaging, and the lateral pixel size was confirmed using an image of a diffraction grating replica specimen as previously described (Chapter 2.2.1.2). SBEM imaging was performed by detection of BSEs using a Zeiss Merlin SEM equipped with a 3View ultramicrotome (Gatan). SBEM imaging was performed by Tom

**Table 4.2. Imaging parameters used during SBEM dataset acquisition.**

CCDBID	ZT	Animal #	Raster Size	Number of Slices	Axial Step Size (nm)	Lateral Pixel Size (nm)
81739	4-6	1	32,000 x 24,000	1,283	30	3.899
90850	16-18	1	32,000 x 24,000	1,604	30	3.899
93678	16-18	2	32,000 x 24,000	1,311	30	3.899

CCDBID	Dwell Time (ns)	Spot Size	Accelerating Voltage (kV)	Magnification
81739	500	1.0	1.9	800
90850	500	1.0	2.2	800
93678	500	1.0	1.9	800

Deerinck in collaboration with Monica Berlanga and myself. A dataset from the second animal of the ZT4-6 time point was collected by Keun-Young Kim and will be used in future analyses (ZT4-6 animal #2: CCDBID 5215795).

## 4.2.2. The subcellular chromorphology of SCN neurons

### 4.2.2.1. Nuclei and nucleoli

All datasets were converted to the MRC format, translationally aligned, and downsampled to isotropic voxels using the methods described in Chapter 2.2.1.3. Training images and labels for nuclei and nucleoli from each dataset were generated by manual segmentation as described in Chapter 2.2.2.2. All training sets had pixel dimensions of 500 x 500 x 50, and individual training slices were taken from points scattered throughout the image stack (Figure 2.8). For each set of training data, a CHM pixel classifier was trained with two stages and two levels as described in Chapter 2.2.2.3. Since isotropic voxels were used, the MPAS algorithm was employed as described in Chapter 3.2.1. The resultant, full-stack probability maps were generated by taking the pixel-by-pixel geometric mean of  $P_{XY}$ ,  $P'_{XZ}$ , and  $P'_{YZ}$ . Segmentations were produced using the automatically-seeded active contour algorithm described in Chapter 2.2.2.5. Nuclear segmentations

were produced using the following values:  $G = 2$ ,  $\alpha = 300$ ,  $\lambda = 8$ . Nucleolar segmentations were produced using the following values:  $G = 2$ ,  $\alpha = 90$ ,  $\lambda = 10$ . An overlap of 50 pixels was used during the tiling and stitching process for all datasets. Surface renderings were produced from each segmentation using the scripts presented in Chapter 3.2.3. A post-rendering size exclusion filter was applied to reject erroneously small or large objects using a custom script. The results for each automatic segmentation were manually inspected for accuracy and corrected when necessary. Metrics of nuclear and nucleolar morphology and spatial orientation were calculated automatically using the method of Chapter 3.2.4.1. Advanced metrics of nuclear topology were calculated using the scripts outlined in Chapter 3.2.4.2. Output CSV files were imported to Microsoft Excel and used for subsequent analyses.

#### **4.2.2.2. Mitochondria**

The converted and aligned datasets produced in the previous section were downsampled by a factor of two in the XY plane using the method presented in Chapter 2.2.1.3. Mitochondrial training sets with pixel dimensions of  $500 \times 500 \times 50$  were produced by manual segmentation. Mitochondria from neurons, astrocytes, and the neuropil were all contained in these training data. This is of particular note because mitochondria from each of these distinct localizations display slightly different textures and grayscale levels. CHM pixel classifiers were trained with two stages and two levels. Pixel classification was performed using a PANFISH implementation of the methods of Chapter 2.2.2.3 developed by Christopher Churas (Churas et al., 2013). Since these datasets were not isotropic, the MPAS algorithm was not used. Segmentations were achieved using the automatically-seeded active contour method with the following values:  $G = 3$ ,  $\alpha = 80$ ,  $\lambda = 7$ . An overlap of 20 pixels was used for the tiling of all datasets during pixel classification. Surface



renderings were produced from each segmentation using the scripts presented in Chapter 3.2.3. Neurons from each dataset were manually segmented in their entirety, and mitochondria were automatically separated to the appropriate neuron using the methods of Chapter 3.2.5. These data are being used to drive the development of an automatic workflow for the analysis of mitochondrial morphology and distribution throughout cells.

#### **4.2.2.3. Stigmoid bodies**

Qualitative inspection of the SBEM datasets of the SCN revealed an unusual neuronal cytoplasmic structure that resembled a nucleolus. Despite initial confusion as to what the structure was, a literature search revealed it to be an organelle known as the stigmoid body (STB). The STB is a membrane-free, proteinaceous inclusion found in the cytoplasm of neurons in many brain regions, including the thalamus, hippocampus, amygdala, and hypothalamus (Shinoda et al., 1992). At the ultrastructural level, the STB resembles a spherical distribution of closely packed clusters of heterogeneous electron-dense granules and fibrils interspersed with seemingly empty pockets (Shinoda et al., 1993). As a whole, STBs are very highly spherical, possess diameters in the range of 0.5-4  $\mu\text{m}$ , and are significantly less electron dense than the nucleolus. Sheets of ER and polyribosomes are frequently seen in the immediate vicinity of STBs (Gutekunst et al., 2003). Reports of cytoplasmic inclusion bodies can be found throughout the history of microscopy in the neurosciences, and structures very closely resembling STBs have been given a plethora of unique names, including nematosomes (Le Beux, 1972; Hindelang-Gertner et al., 1974), botryosomes (Kind et al., 1997), cytoplasmic bodies (Weakley, 1969), giant granular filamentous bodies (Blazquez et al., 1995), and nucleolus-like bodies (Kessel, 1969; Kishi, 1972; Hindelang-Gertner et al., 1974; Takeuchi and Takeuchi, 1982).

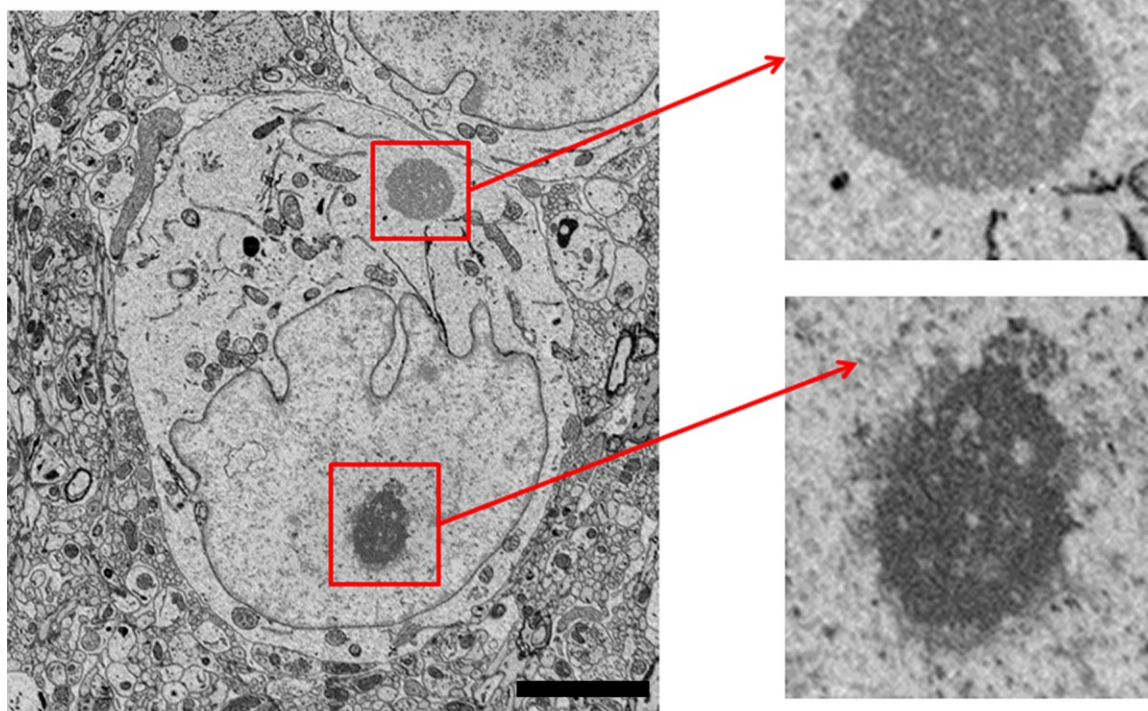
To maintain consistency with what is perceived to be the current standard in the community, the name stigmoid body will be used throughout the rest of this dissertation.

Though not much is known about the function of STBs, it has been hypothesized that they play a protective role against Huntington's disease (HD) by sequestering huntingtin and the androgen receptor, two causative proteins associated with HD (Metzger et al., 2008; Fujinaga et al., 2009). An interacting partner of huntingtin, Huntingtin-associated protein-1 (HAP1), has been identified as a constituent of STBs and is commonly used as a marker for antibody labeling (Gutekunst et al., 1998; Li et al., 1998a). HAP1 has been shown to play a role in neurite outgrowth (Li et al., 2000), and also influences microtubule trafficking by directly binding to motor proteins (Li et al., 1998b). As a consequence of this, Fujinaga and colleagues demonstrated that STB formation is microtubule-dependent and occurs in a two-step process *in vitro* (Fujinaga et al., 2009). The first step, for which microtubules are not necessary, involves the formation of small, HAP1-positive, STB-like inclusions. In the second step, these small inclusions fuse together in a microtubule-dependent process to form spherical, mature STBs. Subsequent studies have demonstrated that the apolipoprotein E receptor, SorLA/LR11, and sortilin co-localize with HAP1 in STBs (Motoi et al., 1999; Gutekunst et al., 2003). Such data indicate that STBs are heterogeneous in both ultrastructure and protein constituency.

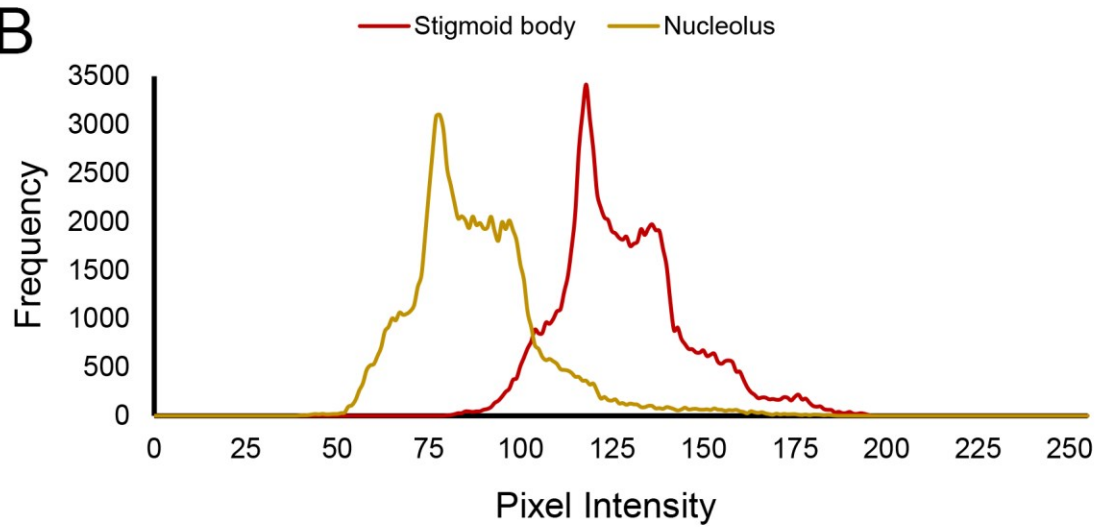
Since the function of STBs largely remains a mystery and no large-scale studies of STB ultrastructure have been performed, the morphology and distribution of STBs in the SCN were studied. The number of SCN neurons containing STBs was determined for each dataset. All STBs were then manually segmented using IMOD. Automatic segmentation of STBs was not attempted because they are rather easy to find and segment manually. The volume and surface area of each STB were computed using the

**Figure 4.3. Stigmoid bodies are cytoplasmic inclusions found in SCN neurons.** Stigmoid bodies are roughly spherical and resemble the nucleolus in terms of texture and size (A, scale bar = 2  $\mu\text{m}$ ). Expanded views of the boxed regions depict a stigmoid body and nucleolus. Nucleoli are, in general, more electron dense than stigmoid bodies. To demonstrate this discrepancy, histograms of pixel intensities are shown for the center of a cross-section through a stigmoid body and nucleolus (B).

A



B



IMOD program *imodinfo*.

### **4.2.3. Electron tomography of SCN organelles**

#### **4.2.3.1. Serial section electron tomography of the stigmoid body**

The same SCN tissues blocks used for SBEM imaging were subsequently utilized to create grids of serial thick sections for ssET. Ribbons of 250 nm serial sections were cut using an ultramicrotome and collected on Luxel slot grids. Five grids containing ribbons of 4-5 sections each were prepared. All grids were carbon coated and glow discharged, and a 50:50 solution of 5% bovine serum albumin (BSA) and 15/20 nm colloidal gold particles (AuNPs) was applied to each grid. No poststaining was performed. This process was repeated for the tissue of one animal from both time points.

Microscopy was performed using a JEOL 4000EX IVEM operating at 300 kV. Samples were loaded using a rotation holder to permit the collection of tilt series about a second axis without the need to remove the holder from the column. Low magnification maps of a STB across multiple sections were recorded, and it was confirmed that the entire STB was contained and undamaged in the serial series before imaging. Tilt series were acquired at 30,000x magnification, corresponding to a lateral pixel size of 0.497 nm. All tilt series were acquired in 1° angular increments, and most were acquired with the tilt range of  $\pm 60^\circ$ . The SerialEM software package (Mastronarde et al., 2003) was used to automatically track and maintain focus between successive tilts. Each tilt series was recorded in two steps: (1) 0 to  $-60^\circ$  and (2) 0 to  $+60^\circ$ . Following the collection of tilt series for all sections of each grid, the grid was rotated by  $90^\circ$  and second tilt series were acquired about the same region of each grid.

Two serial series were acquired using the above procedure. The first was a STB from a ZT16-18 neuron and consisted of data for ten 250 nm sections. The second was a

STB from a ZT4-6 neuron and consisted of data for six 250 nm sections. Tilt series were processed and aligned, and tomographic reconstructions were computed using the Etomo GUI of IMOD as previously described (Kremer et al., 1996; Perkins et al., 2009). Serial sections have not yet been stacked, but will be in the future.

#### 4.2.3.2. Electron tomography of neuronal nuclei

Multiple high magnification tilt series of neuronal nuclei were acquired from the same grids prepared in the previous section. Single, non-serial tilt series were collected from both ZT4-6 and ZT16-18 grids. A list of the tilt series acquired and their corresponding imaging parameters is given in Table 4.3. Tomographic reconstruction was performed as described in the previous section.

**Table 4.3. Imaging parameters used for the collection of tomographic tilt series of SCN nuclei**

	Feature Imaged	CCDBID	Section Thickness (nm)	Accelerating Voltage (kV)	Pixel Size (nm)	Angular Min, Max, Increment
ZT4-6	NE	66441	500	400	0.49	-58,+58,2
ZT4-6	NE	66526	500	400	0.49	-56,+56,2
ZT4-6	NE	66610	500	400	0.92	-50,+54,2
ZT4-6	NE Invagination	66971	500	400	0.49	-54,+54,2
ZT4-6	NE	67132	500	400	0.49	-58,+48,2
ZT4-6	NE	67833	500	400	0.49	-50,+58,2
ZT16-18	NE	69548	250	300	0.33	-58,+58,2
ZT16-18	NE Invagination	69011	250	400	0.62	-58,+58,2
ZT16-18	NE Invagination	69459	250	400	0.62	-58,+58,2

### 4.3. Results

#### 4.3.1. Nuclear volume, surface area, and topology

Automatically generated segmentations appeared consistent in all three datasets. Due to the extremely labor intensive nature of generating ground truth for comparison, segmentation evaluation metrics were not computed for the ZT16-18 datasets. The segmentation evaluation metrics for the ZT4-6 dataset were reported in Chapter 2 (Table 2.3; Figure 2.15). The results of the morphological characterization of SCN neuronal nuclei from all three SBEM datasets are reported in Table 4.4. Only nuclei that were fully contained in the dataset were included in the analysis. Histograms of nuclear volume and surface area are given in Figures 4.4 and 4.5, respectively. A graphical representation comparing the advanced topological metrics for each dataset is shown in Figure 4.6. Tomographic reconstruction demonstrate the presence of NPCs and membranous debris or vesicles in the lumen of nuclear invaginations (Figure 4.7).

**Table 4.4. The results of nuclear morphological characterization for three SCN SBEM datasets.** Metrics were automatically computed using the methods previously described. All values are reported as the mean  $\pm$  standard deviation.

	ZT4-6, Animal #1	ZT16-18, Animal #1	ZT16-18, Animal #2
<b>Count</b>	81	72	96
<b>Volume (<math>\mu\text{m}^3</math>)</b>	331.51 $\pm$ 32.84	305.07 $\pm$ 31.19	287.60 $\pm$ 29.51
<b>Surface Area (<math>\mu\text{m}^2</math>)</b>	316.13 $\pm$ 38.09	289.74 $\pm$ 37.27	300.79 $\pm$ 46.33
<b>Surface Area to Volume Ratio</b>	0.957 $\pm$ 0.107	0.951 $\pm$ 0.093	1.043 $\pm$ 0.091
<b>Invagination Factor</b>	1.367 $\pm$ 0.143	1.322 $\pm$ 0.130	1.423 $\pm$ 0.147
<b>Shape Index</b>	0.377 $\pm$ 0.052	0.446 $\pm$ 0.059	0.381 $\pm$ 0.050
<b>Convex Hull Difference (%)</b>	4.396 $\pm$ 2.195	3.026 $\pm$ 1.747	4.973 $\pm$ 2.075
<b>Sphericity</b>	0.740 $\pm$ 0.077	0.763 $\pm$ 0.074	0.710 $\pm$ 0.072

### 4.3.2. Nucleolar number and volume

The proposed workflow for automatic analysis accurately separated all nucleoli into their proper nuclei, as confirmed by visual inspection. Histograms of the computed total neuronal nucleolar volume per cell are displayed in Figure 4.7 for all three datasets. The mean total nucleolar volume was  $3.18 \pm 0.64 \mu\text{m}^3$  for the ZT4-6 dataset and  $3.03 \pm 1.03 \mu\text{m}^3$  and  $3.04 \pm 0.65 \mu\text{m}^3$  for the two ZT16-18 datasets. In addition to total volume, the number of neurons containing one, two, or greater than two nucleoli is also compared in Figure 4.6. Fifty-seven percent of neurons in the ZT4-6 dataset and 69% and 84% of neurons in the ZT16-18 datasets contained just one nucleolus. Nucleoli from cells containing a single nucleolus were clustered and morphologically compared to nucleoli from cells containing multiple nucleoli. The mean nucleolar volume and distance to the nuclear centroid were computed for each group and the results are reported in Table 4.5. Histograms of the nucleolar volume fraction are shown in Figure 4.8 for all datasets.

### 4.3.3. Stigmoid body number and morphology

Histograms demonstrating the frequency of STB volumes are given in Figure 4.10 for each dataset. The percentage of neurons containing zero, one, and two STBs in each dataset are also reported in Figure 4.10. Average STB shape was assessed by computing the sphericity and invagination factor for every STB analyzed. The mean values of these metrics for each dataset are shown in Figure 4.11. As expected, STBs were uniformly very close to spherical in all datasets. Some STBs containing tunnels were found in SBEM datasets (Figure 4.12). Such tunnels were commonly associated with ER, and the ER network could frequently be tracked from one end of the tunnel to the other. The lumens of these tunnels possess electron densities similar to those of the cytoplasm. Other



**Table 4.5. Nucleolar volume and positioning in SCN neurons with single and multiple nucleoli.** All values are reported as the mean  $\pm$  standard deviation.

	ZT4-6, Animal #1		ZT16-18, Animal #2		ZT16-18, Animal #2	
	Single	Multiple	Single	Multiple	Single	Multiple
<b>Number of Nucleoli (N)</b>	46	80	50	49	80	37
<b>Nucleolar Volume (<math>\mu\text{m}^3</math>)</b>	3.26 $\pm$ 0.61	1.36 $\pm$ 1.88	3.17 $\pm$ 0.94	1.21 $\pm$ 0.79	3.04 $\pm$ 0.61	1.24 $\pm$ 0.86
<b>Distance to Nuclear Centroid (<math>\mu\text{m}</math>)</b>	3.62 $\pm$ 1.66	6.10 $\pm$ 2.47	3.39 $\pm$ 1.46	5.03 $\pm$ 2.49	3.71 $\pm$ 2.01	5.91 $\pm$ 2.29

membrane-bound organelles, such as mitochondria and lysosomes, were also found in close proximity to STBs.

#### 4.4. Discussion

The results of this chapter demonstrate the utility of the automatic segmentation and quantification workflow proposed in Chapters 2 and 3. This workflow was used to quantify the morphologies of SCN neuronal nuclei and nucleoli in three different, large-scale SBEM datasets. Accurate results were achieved in all datasets. Mitochondrial segmentations have also been generated for all datasets described in this chapter, though the mitochondrial morphologies from these segmentations remain to be analyzed in future work. Some preliminary results obtained from this effort will be presented and discussed in Chapter 5.

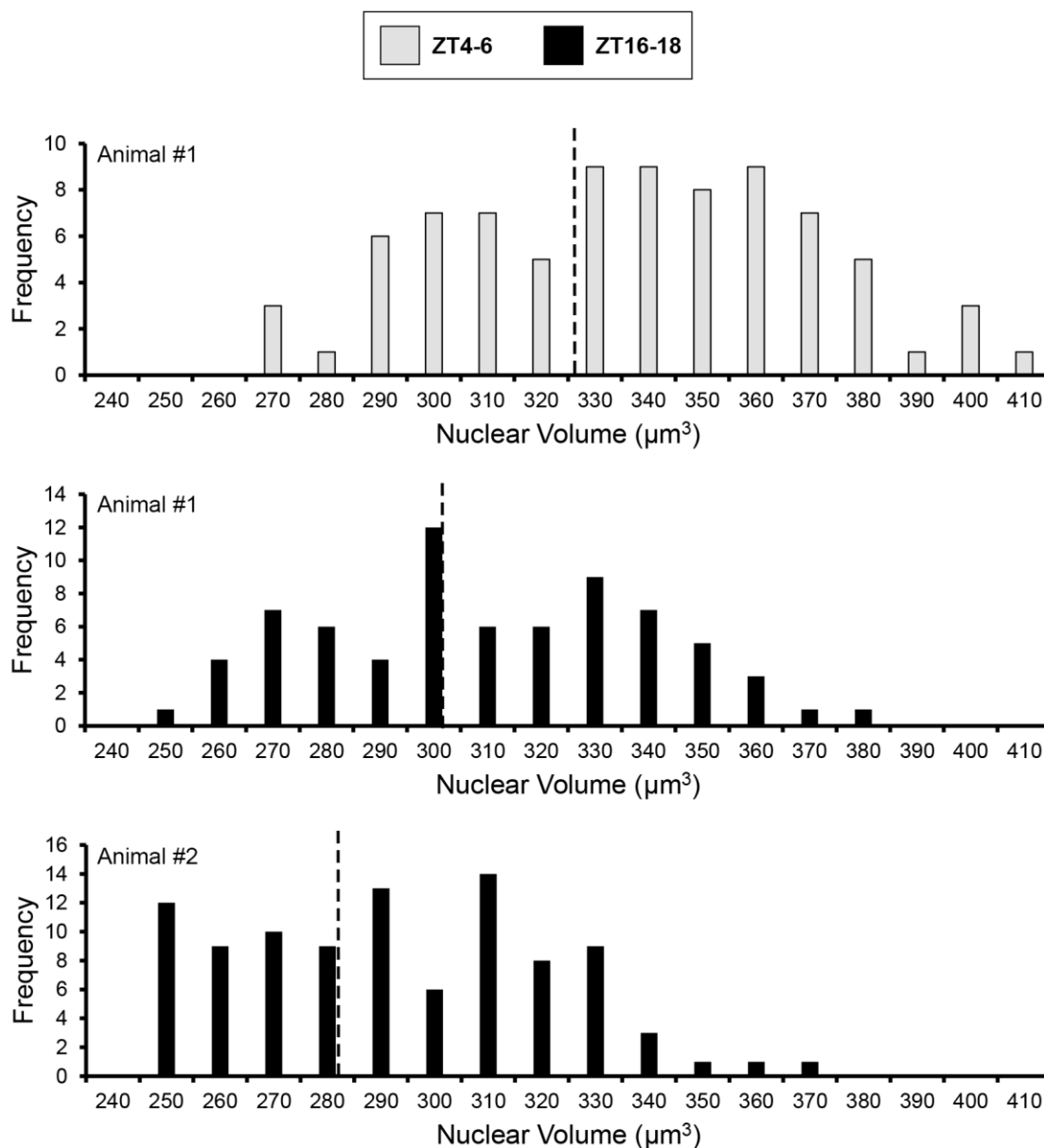
The data reported here demonstrate a trend of increased nuclear volume during the middle of the light phase (ZT4-6), which corresponds to the period of inactivity of the nocturnal mouse (Figure 4.4). Metrics designed to quantify the extent of nuclear invagination do not show obvious variations in their average values across the three datasets analyzed (Figure 4.6). This is also supported by qualitative observation; nuclei in

all three datasets appear to be heavily invaginated. Volumetric increases across the diurnal cycle may be facilitated by the transient remodeling of the NE/ER system (Lammerding et al., 2007). However, since only one light phase dataset and two dark phase datasets were analyzed, these data are preliminary. More datasets will be analyzed in the future to increase the sample size and facilitate statistical analyses.

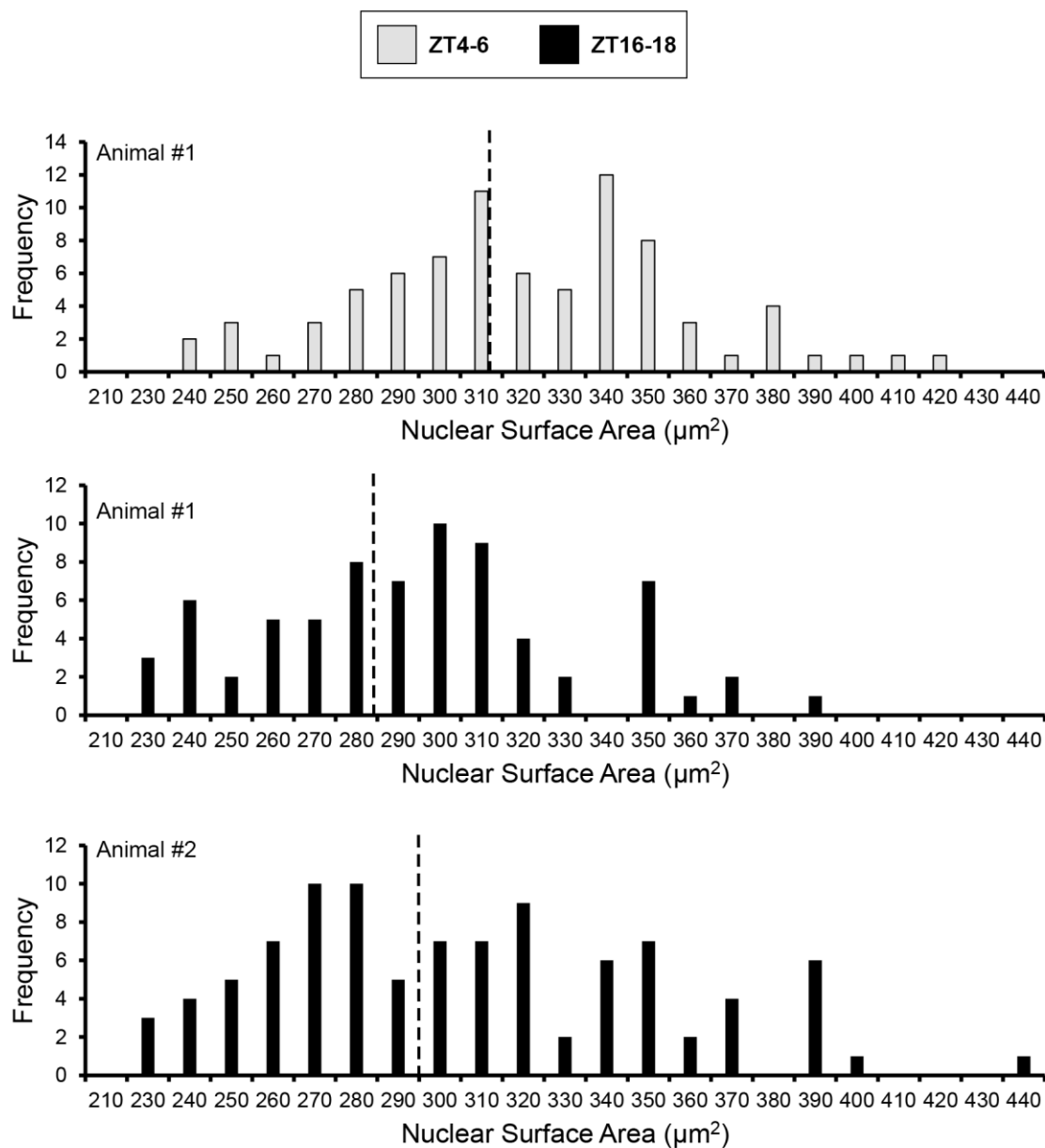
The total nucleolar volume and nucleolar volume fraction demonstrate remarkably similar averages and distributions for neurons in the three datasets analyzed (Figures 4.8 and 4.9). The data of Figure 4.8 suggest that there may be a greater percentage of neurons with multiple nucleoli during the dark cycle, which corresponds to the period of activity for the animal. Nucleoli also occupy consistently similar positions within the nucleus in all datasets studied (Table 4.5). These observations will also need to be confirmed by the analysis of additional datasets.

Though stigmoid bodies have been previously identified in the SCN (Shinoda et al., 1992), this was the first known instance in which they were studied ultrastructurally using 3D EM. STBs in the SCN demonstrate a wide distribution of sizes (Figure 4.10) and are highly spherical (Figure 4.11). Many of the STBs included in this analysis were clearly smaller and ultrastructurally different than the canonical 1-4  $\mu\text{m}$  diameter STB (Shinoda et al., 1993). Such STBs did not possess pockets, tunnels, or obvious associations with cytoplasmic organelles. It is possible that these small STBs are the precursors to mature STBs that have been reported *in vitro* (Fujinaga et al., 2009). The percentage of SCN neurons containing STBs was highly variable across all three datasets (Figure 4.10). Therefore, STB presence in neurons may vary by individual rather than being associated with any circadian cycle. However, it is interesting to note that the vast majority of neurons in the single light cycle dataset did not contain STBs. Data from more animals will need to be analyzed to establish if this trend holds.

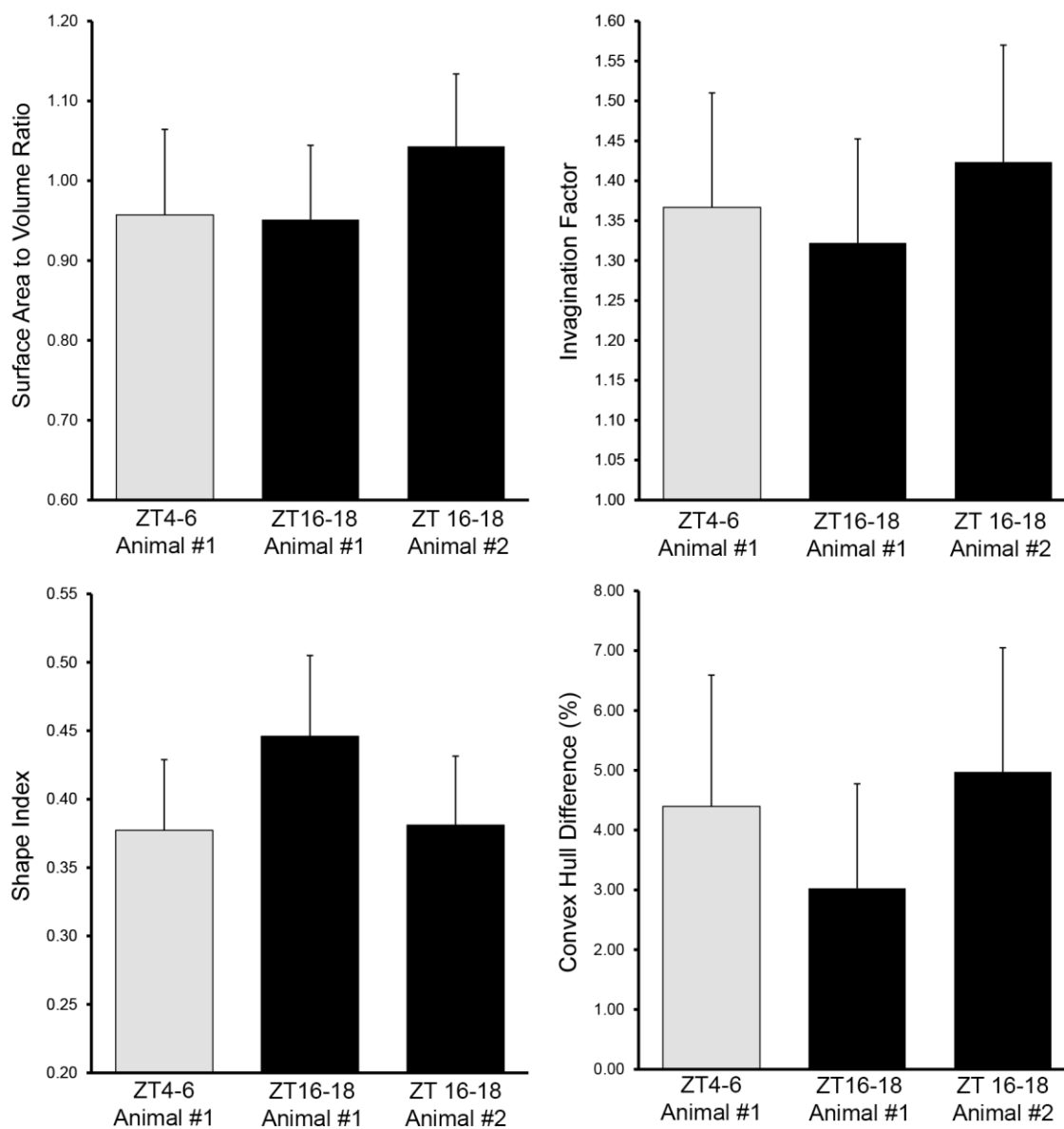
In conclusion, though the data presented here are preliminary, the results of this chapter demonstrate the power of automated SBEM analyses for the quantification of organelle morphologies. Such a study provides far greater insight into 3D organelle morphology and spatial organization than was afforded by the previous chromomorphological studies discussed in Chapter 4.1.2. In future analyses, it may be advisable to collect smaller SBEM datasets from more animals. Such an experimental design would facilitate the acquisition of statistically verifiable data and decrease the computational demand required for automatic segmentation.



**Figure 4.4. The distribution of nuclear volumes in SCN neurons.** The volumes of automatically segmented nuclei were computed using the workflow for morphological characterization reported in Chapter 3.2.4. Shown here are histograms illustrating the spread of nuclear volumes for SCN neurons from the three datasets analyzed. The mean nuclear volume is indicated by a dashed line on each histogram (ZT4-6 Animal #1:  $331.51 \pm 32.85 \mu\text{m}^3$ ,  $N = 81$ ; ZT16-18 Animal #1:  $305.07 \pm 31.19 \mu\text{m}^3$ ,  $N = 72$ ; ZT16-18 Animal #2:  $287.60 \pm 29.51 \mu\text{m}^3$ ,  $N = 96$ ; values are reported as the mean  $\pm$  standard deviation).

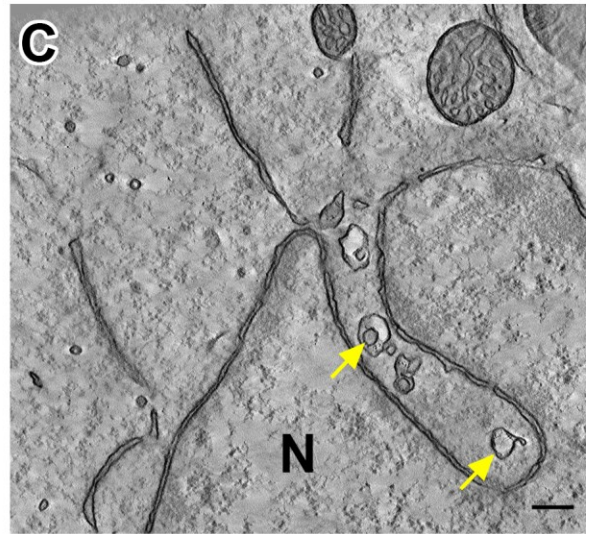
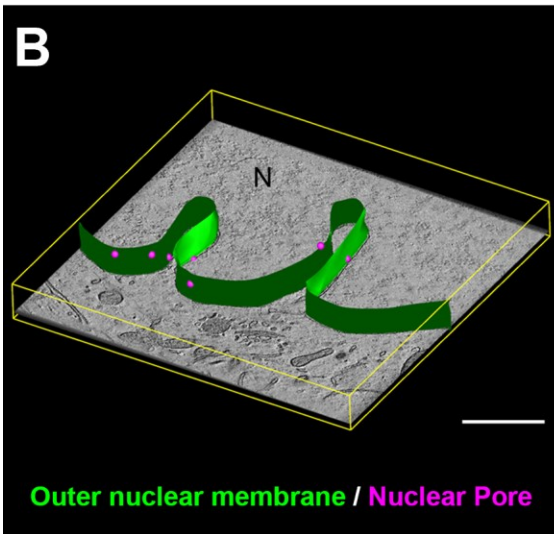
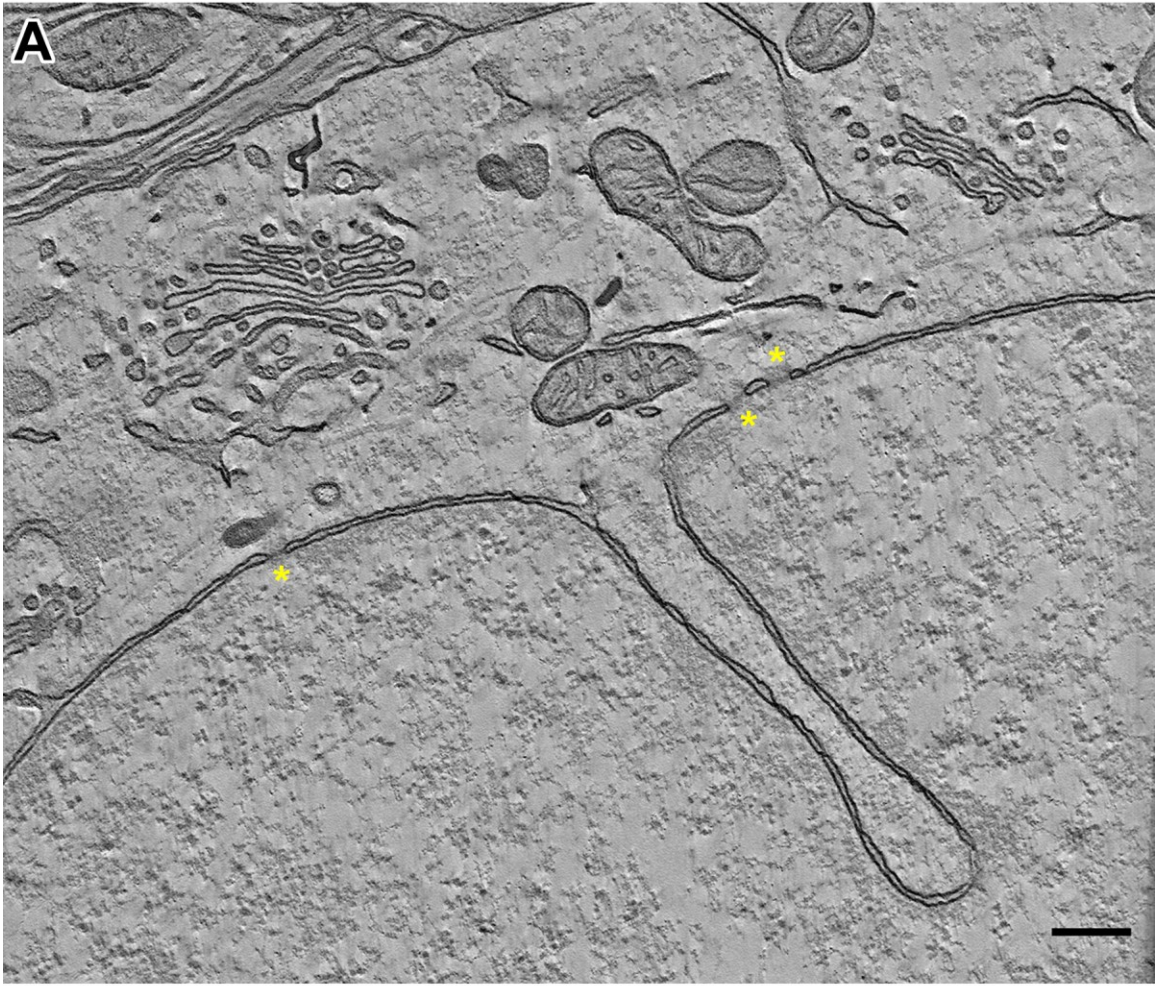


**Figure 4.5. The distribution of nuclear surface areas in SCN neurons.** The surface areas of automatically segmented nuclei were computed using the workflow for morphological characterization reported in Chapter 3.2.4. Shown here are histograms illustrating the spread of nuclear surface areas for SCN neurons from the three datasets analyzed. The mean nuclear surface area is indicated by a dashed line on each histogram (ZT4-6 Animal #1:  $316.13 \pm 38.08 \mu\text{m}^2$ ,  $N = 81$ ; ZT16-18 Animal #1:  $289.737 \pm 37.27 \mu\text{m}^2$ ,  $N = 72$ ; ZT16-18 Animal #2:  $300.79 \pm 46.33 \mu\text{m}^2$ ,  $N = 96$ ; values are reported as the mean  $\pm$  standard deviation).



**Figure 4.6. The mean values of topological descriptors for nuclear invagination in nuclei of SCN neurons.** The mean values of the surface area to volume ratio, invagination factor ( $IF_{3D}$ ), shape index ( $\sigma$ ), and convex hull difference (CHD) are reported for nuclei of SCN neurons from the three datasets analyzed. Error bars represent the standard deviation. The mean numerical values of each metric are given in Table 4.4.

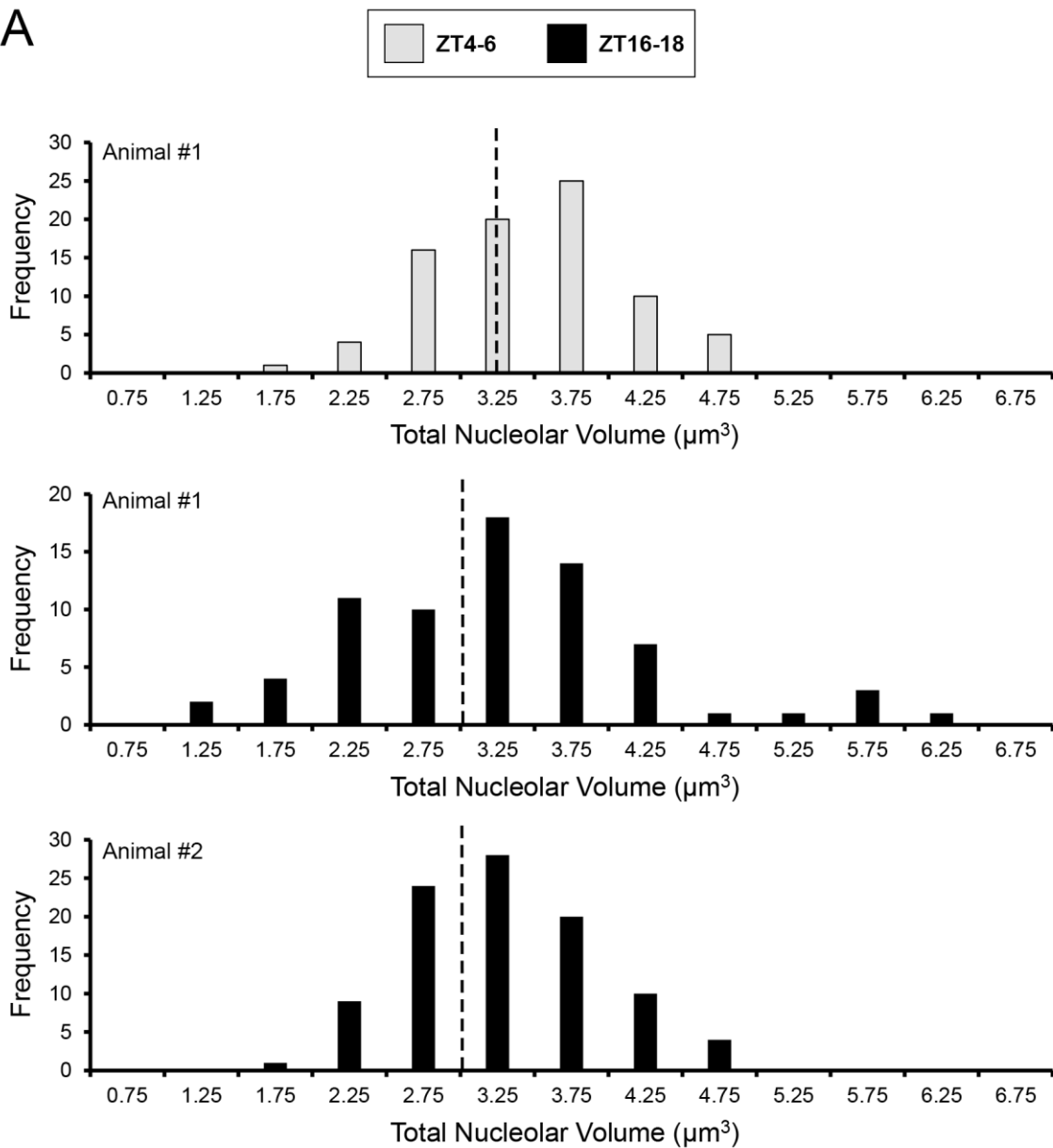
**Figure 4.7. Electron tomography of SCN nuclei reveals the ultrastructural characteristics of their nuclear invaginations.** The results of tomographic reconstruction are shown here. A single slice through a tomogram depicts a nuclear invagination with cytoplasmic organelles in its vicinity (A). The lumen of the invagination in this reconstruction is devoid of membrane-bound organelles. As previously reported in other tissue types, SCN invaginations contain NPCs (B). A slice through another tomogram demonstrates the presence of membranous debris or large vesicles in the lumen of the invagination (C). The nature of these structures is unclear. (A: scale bar = 200 nm; B: scale bar = 500 nm; C: scale bar = 200 nm).



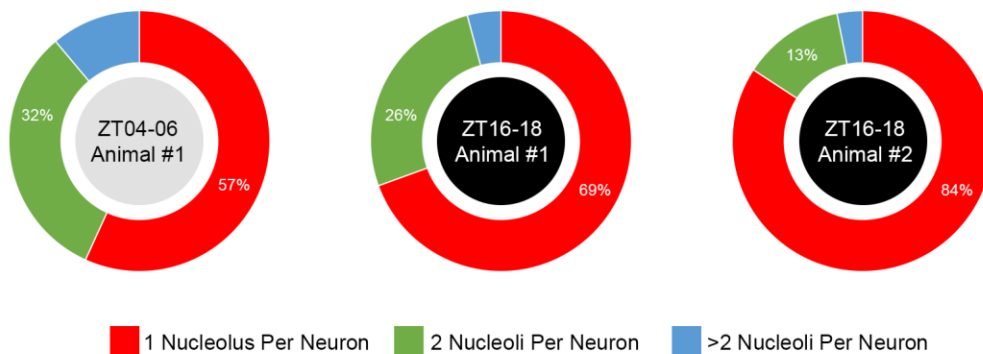


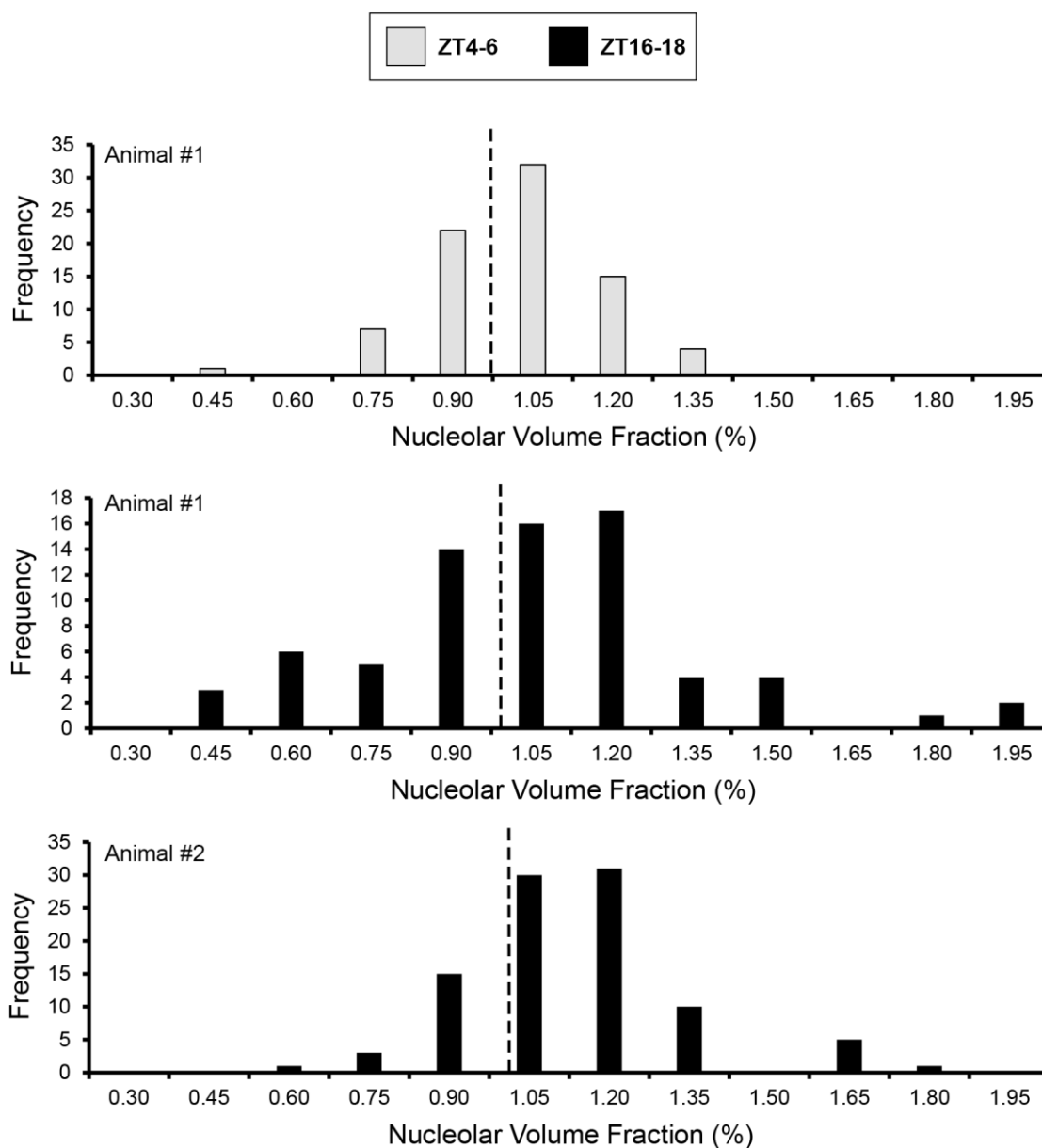
**Figure 4.8. The distribution of total nucleolar volume in SCN neurons and the percentage of neurons containing multiple nucleoli.** (A) The volumes of automatically segmented nucleoli were computed using the workflow for morphological characterization reported in Chapter 3.2.4. Shown here are histograms illustrating the spread of total nucleolar volume per SCN neuron from the three datasets analyzed. The mean total nucleolar volume is indicated by a dashed line on each histogram (ZT4-6 Animal #1:  $3.18 \pm 0.64 \mu\text{m}^3$ , N = 81; ZT16-18 Animal #1:  $3.03 \pm 1.03 \mu\text{m}^3$ , N = 72; ZT16-18 Animal #2:  $3.04 \pm 0.65 \mu\text{m}^3$ , N = 96; values are reported as the mean  $\pm$  standard deviation). (B) The percentages of nuclei containing one nucleolus, two nucleoli, or greater than two nucleoli are depicted graphically.

A



B

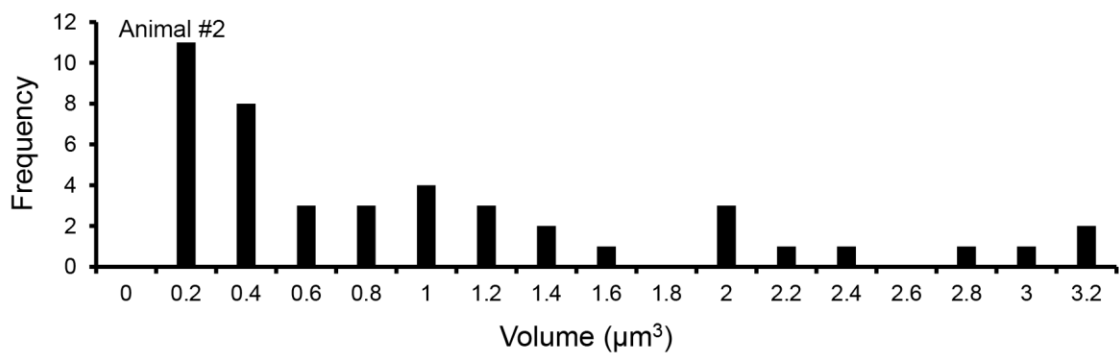
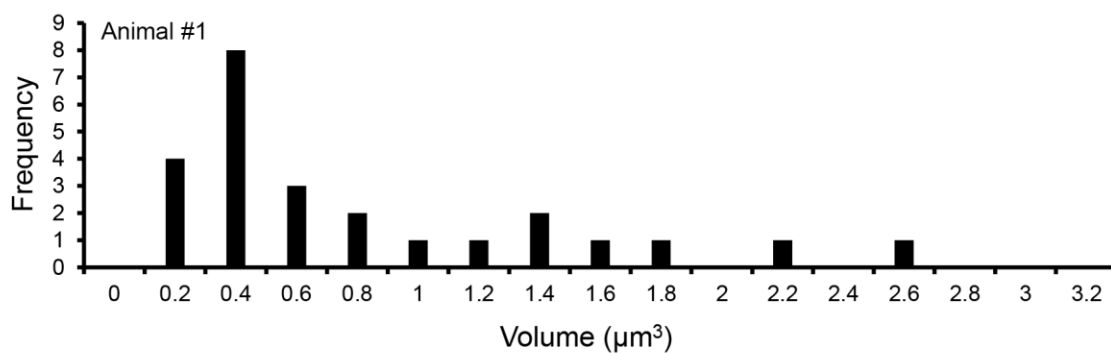
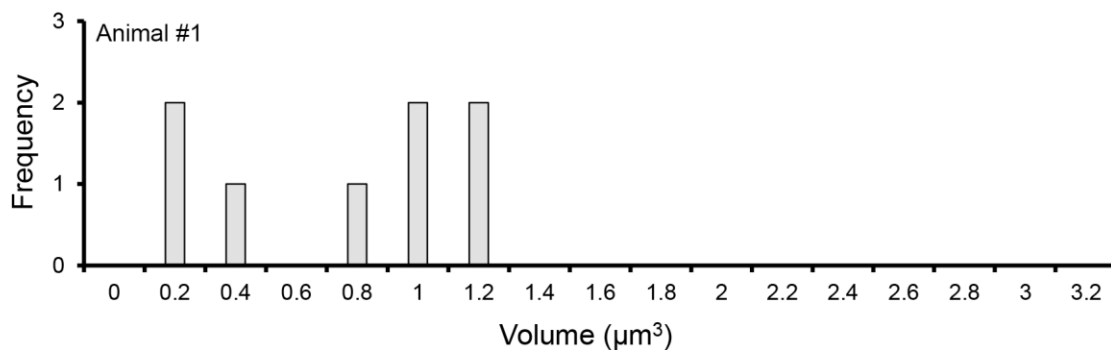




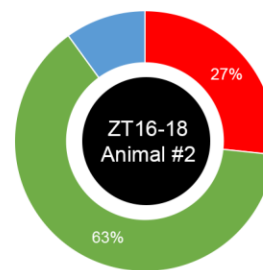
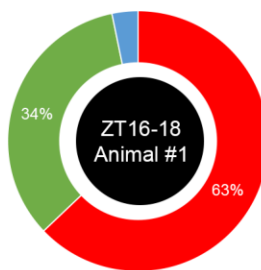
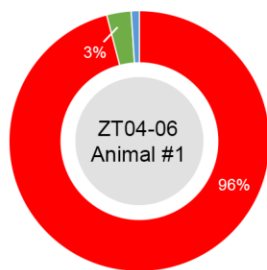
**Figure 4.9. The distribution of nucleolar volume fraction in SCN neurons.** The nucleolar volume fraction, calculated as the ratio of nucleolar volume to nuclear volume, was calculated for each SCN neuron from the automatically segmented data. Shown here are histograms illustrating the spread of nucleolar volume fraction from the three datasets analyzed. The mean nucleolar volume fraction is indicated by a dashed line on each histogram (ZT4-6 Animal #1:  $0.95 \pm 0.15\%$ ,  $N = 81$ ; ZT16-18 Animal #1:  $0.98 \pm 0.31\%$ ,  $N = 72$ ; ZT16-18 Animal #2:  $1.05 \pm 0.20\%$ ,  $N = 96$ ; values are reported as the mean  $\pm$  standard deviation).

**Figure 4.10. The distribution of stigmoid body volume in SCN neurons and the percentage of neurons containing stigmoid bodies.** (A) The volumes of manually segmented stigmoid bodies were computed. Shown here are histograms illustrating the spread of stigmoid body volumes from the three datasets analyzed. (B) The percentages of SCN neurons containing zero, one, and two stigmoid bodies per soma are depicted graphically.

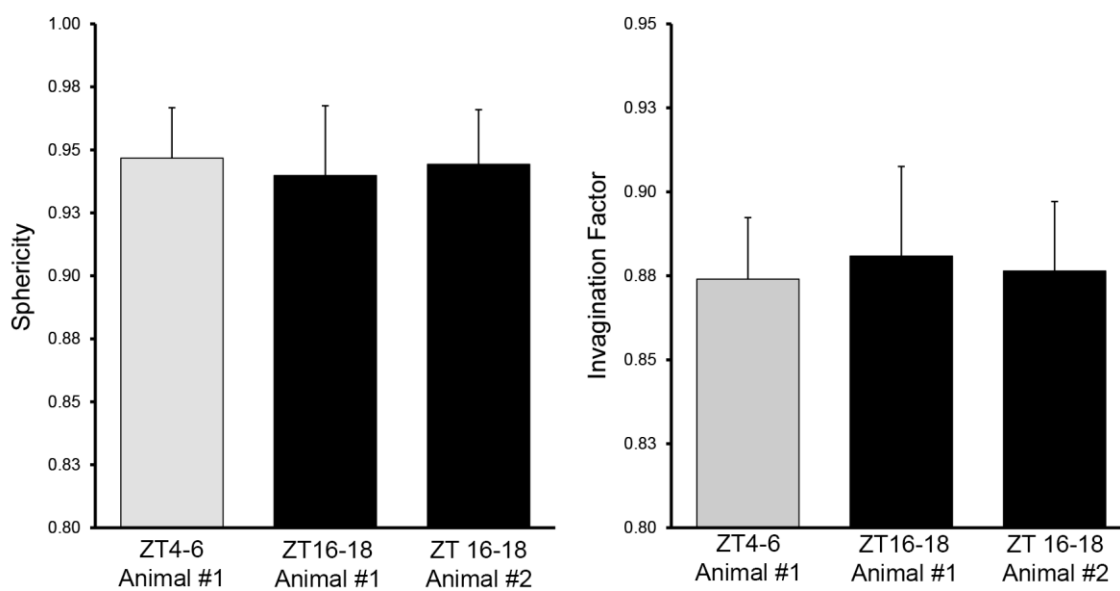
A



B



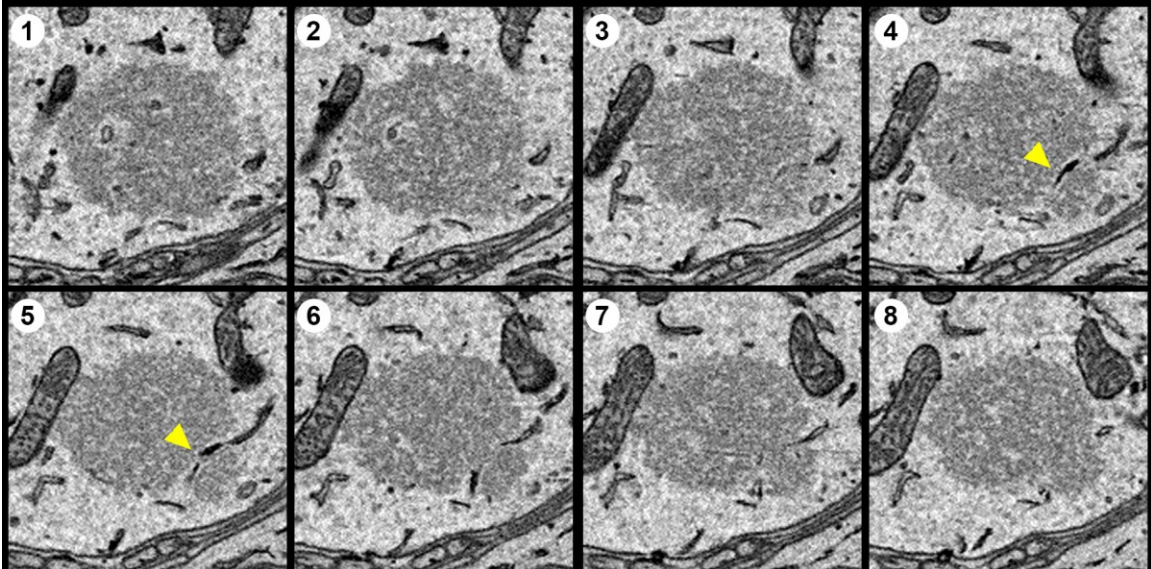
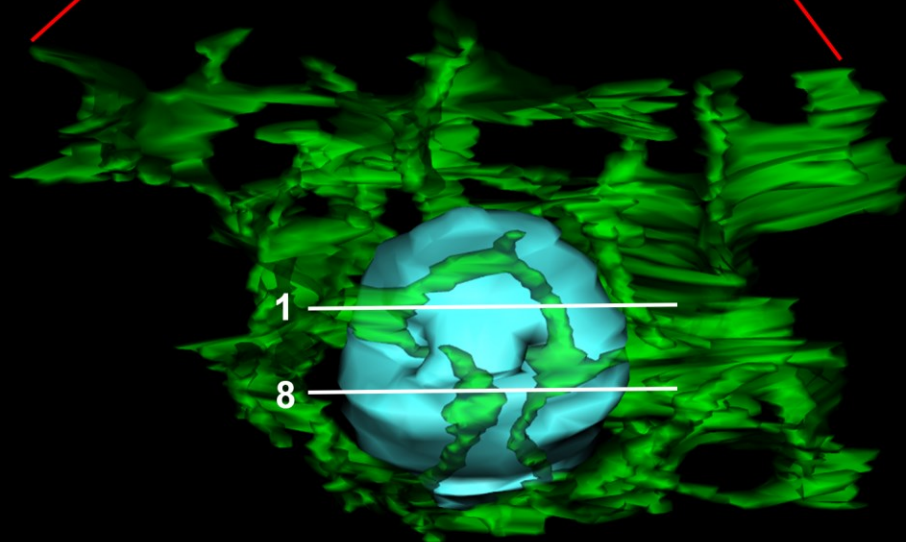
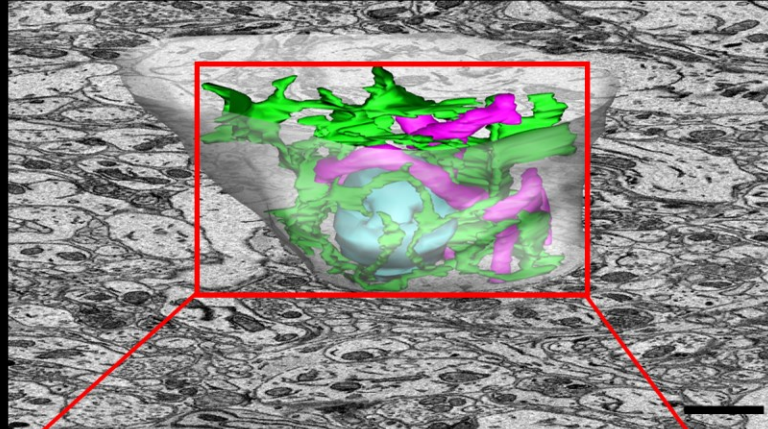
■ 0 Stigmoid Bodies Per Soma   
 ■ 1 Stigmoid Body Per Soma   
 ■ 2 Stigmoid Bodies Per Soma



**Figure 4.11. The mean values of topological descriptors for stigmoid bodies in SCN neurons.** The mean values of the sphericity and invagination factor ( $IF_{3D}$ ) are reported for stigmoid bodies in SCN neurons from the three datasets analyzed. Error bars represent the standard deviation. The sphericity and invagination factor are exactly one for a perfect sphere; therefore, these data demonstrate that stigmoid bodies are remarkably spherical.

**Figure 4.12. Stigmoid bodies contain tunnels associated with the endoplasmic reticulum.** In addition to having numerous organelles such as mitochondria and lysosomes in their vicinity, STBs sometimes contain short tunnels by which ER crosses from one side of the STB to the other. Shown here (top, scale bar = 1  $\mu\text{m}$ ) is a manual reconstruction of the vicinity of a single STB (cyan) from a ZT16-18 SCN dataset. Mitochondria (magenta), ER (green), and the surrounding plasmalemma (translucent white) were also reconstructed. The STB is located at the extremity of a neuronal soma, in close proximity to the axon hillock. The boxed region is magnified in the second rendering (middle), which shows only the STB and its surrounding ER network. Part of the ER network traverses through a tunnel in the STB and emerges from the other end. The white lines on this rendering illustrate the top and bottom boundaries of the eight corresponding and sequentially numbered SBEM slices shown below. All slices were spaced 30 nm apart from one another. The ER tunnel is clearly visible in slices four and five (yellow arrowheads). The lumen of the tunnel appears to have a similar electron density to that of the cytoplasm. Electron tomograms of STBs also demonstrate the presence of membranous structures in what appears to be tunnels (data not shown), though the function of such tunnels remains unclear.

Stigmoid Body / Endoplasmic Reticulum / Mitochondria





## **Chapter 5**

### **Conclusions and Future Perspectives**

## 5.1. Contributions, significance, and limitations

Novel technologies such as SBEM, FIBSEM, and array tomography have supplied scientists with the tools necessary to image significant percentages of mammalian neuroanatomical regions at unprecedented scale and resolution. These methods currently facilitate the collection of teravoxels of image data per day, and new instruments promise to push this number significantly higher in the immediate future (Marx et al., 2013; Keller et al, 2014). Unfortunately, as has been the case for decades (Macagno et al., 1979), our ability to analyze such data continues to significantly lag behind our ability to collect it. Furthermore, many algorithmic approaches to automatic image analysis are not readily accessible or easily implementable to the general scientific community. As a result, most EM facilities are currently producing a vast surplus of image data that cannot practically be analyzed without significant breakthroughs from the image processing community.

The methods presented in this dissertation provide a workflow for the automatic analysis of organelles from such data in a reasonable time frame. The work of Chapter 2 supplied a pipeline for the application of supervised machine learning algorithms to the task of automatically segmenting diverse organelle targets in SBEM datasets. An algorithm for the binarization of organelle-specific probability maps based on active contour evolution at automatically seeded points was implemented and described. Performance was assessed by mathematically comparing automatically generated segmentations to manually segmented ground truth, and accurate results were demonstrated for all four organelles tested. The results of this method were subsequently validated by comparison to another recently published, supervised algorithm. Finally, and perhaps most importantly, a method for the computationally efficient scaling of this workflow to teravoxel-sized datasets was presented. The decomposition of input images into small tiles decreases the memory demand for pixel classification and increases the

degree to which the process can be parallelized. Using such parallelization on supercomputing resources, dataset-wide automatic segmentations were achieved in the range of one to seven days. The only required human input to this process is the manual segmentation of training data, which, even in the worst case scenario, requires only a few hours. Considering it would take a single person many years to manually segment all organelles from a similarly sized dataset, the utility of this approach is obvious.

Surprisingly, very few reports of large-scale organelle segmentations of this nature exist in the literature. Furthermore, the use of such data for the reconstruction and quantification of organelle morphologies is even less common. While some dataset-wide automatic segmentations of organelles have been produced (Lucchi et al., 2012; Tek et al., 2014), these results were only used as proof-of-concept validations of their algorithmic approaches. The work of Noske and colleagues produced detailed characterizations of organelle morphologies, but their study was limited to only two cells (Noske et al., 2008). Since organelle structure is known to correlate with a host of normal and pathological processes in biology (Knott, A.B. et al., 2008; Worman, 2012), the generation of large-scale maps of *in situ* organelle structure and distribution across hundreds of cells is highly desirable. The work presented in Chapter 3 addressed this need by introducing a quantitative analysis extension to the segmentation workflow of Chapter 2. First, an accelerated pipeline for generating 3D surface renderings from a stack of 2D segmentations was described. A script capable of automatically computing morphological parameters and spatial distributions from these renderings was developed, and the results of its application to nuclei and nucleoli from numerous datasets are reported in Chapter 4. Importantly, the quantitative analysis methods described in Chapter 3 can be linked directly to the output of automatic segmentations in a seamless workflow to automatically yield morphological metrics.

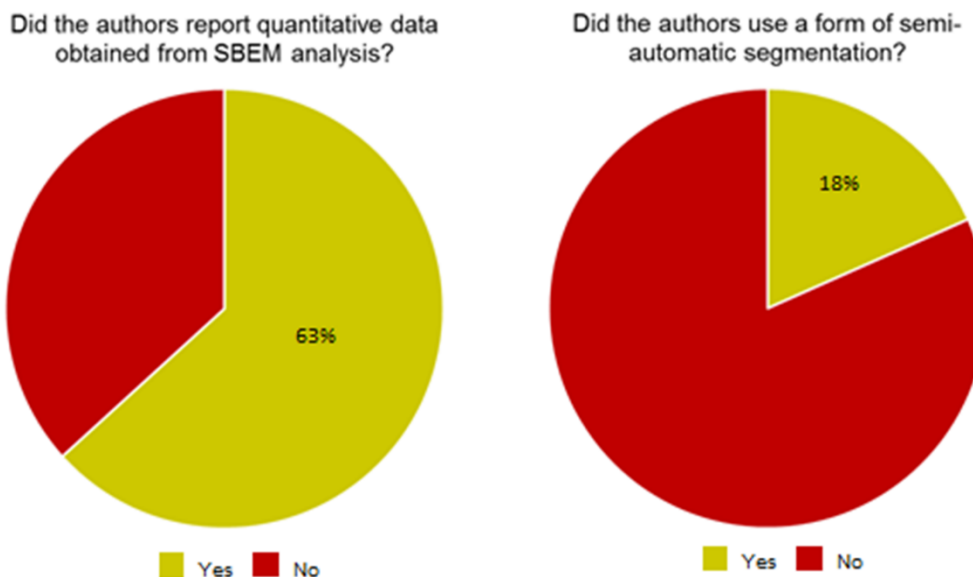
A significant roadblock to the full automation of this process lies in the fact that automatically generated segmentations frequently need to be manually inspected and corrected before the morphologies extracted from them can be deemed accurate. In Chapter 3, two novel methods to decrease the need for manual editing were introduced. The first, the MPAS algorithm, produces representations of organelle morphology that are more accurate by averaging probability maps acquired from multiple views of the same object. It was demonstrated that this method can reduce the errors associated with estimations of nuclear volume and surface area to a few percent without the need for manual corrections. This increased accuracy, of course, comes at the cost of increased computational load. However, this increased load may be justified in cases where MPAS can dramatically reduce the time needed for manual correction by humans. A method for correcting segmentations by replacing poorly segmented slices of objects with their interslice interpolations was also introduced. This method can be used on single-axis probability maps or in tandem with MPAS to improve segmentation accuracy. The next step in its development will be to implement a set of rules for the automatic detection of poorly segmented slices; such rules will likely need to vary on an organelle-by-organelle basis. Taken together, both of these methods facilitate the improvement of segmentation results without the need for user interaction, and should dramatically reduce the time required for manual correction.

## **5.2. Future perspectives**

Data from a single SBEM dataset can be used to pose several interesting questions. However, obtaining answers to these questions typically necessitates the use of manual segmentation, which is untenable at the scale of teravoxel-sized SBEM image stacks (Figure 2.1). Additionally, the application of many existing automatic segmentation

approaches is hindered by a number of factors. First, many published algorithms have been tested on only a single dataset and, in many cases, only a small subset of a single dataset. Therefore, it is unclear if such algorithms will yield satisfactory results when applied to images of different anatomical regions that were acquired using a different instrument and imaging parameters than the published test dataset. Second, many algorithms contain numerous poorly defined variables that significantly affect segmentation performance. Such variables must be tuned to different factors of the input images, and it is frequently unclear how this tuning should be performed. Lastly, many open-source automatic segmentation packages contain numerous and often poorly defined dependencies. The simple act of successfully installing or compiling them on a machine may require several days and significant levels of frustration.

All of these factors have combined to make most automatic segmentation approaches unattractive to the general scientific community. A survey of published studies that employed SBEM from 2006-2014 (Appendix A) revealed that only 63% reported quantitative data derived from SBEM image analyses (Figure 5.1). Furthermore, only 18% of the studies that reported quantitative data employed automatic or semi-automatic segmentation techniques. This is a shockingly low number, and indicates that there is a great need within the field for intuitive and easy-to-apply automatic segmentation algorithms. The most well-known software package designed to address this need is *ilastik* (Sommer et al., 2011), which provides a user-friendly GUI and easy-to-follow tutorials. However, *ilastik* trains classifiers based on 3D rather than 2D features, a fact that makes it better situated to process data from modalities that provide voxel sizes that are close to isotropy, such as FIBSEM. As discussed in detail in Chapter 1, however, SBEM is currently the modality of choice for large-scale studies covering significant tissue



**Figure 5.1. A survey of the SBEM literature revealed that the vast majority of studies did not employ automatic or semi-automatic analyses.** The numbers shown in this figure were generated from the SBEM publications listed in Appendix A, and cover a time span of 2006 through March 2014 (N = 49).

volumes. The desire to segment organelles from SBEM datasets led to the development of the technologies described in this dissertation.

The workflow presented here circumvents many of the classical problems associated with automatic segmentation routines. First, no assumptions about organelle shape or geometry are necessary, which makes the method applicable to any target rather than just one specific organelle type. Second, though certain variables ( $N_{\text{stages}}$ ,  $N_{\text{levels}}$ ,  $G$ ,  $\lambda$ ,  $\alpha$ ) are used, most do not have a dramatic impact on segmentation performance and there is rarely a need to change them. Accurate segmentations for all organelles studied have been achieved using the same parameter set ( $N_{\text{stages}} = 2$ ,  $N_{\text{levels}} = 2$ ,  $G = 2$ ,  $\lambda = 100$ ,  $\alpha = 6$ ). Finally, to allow for widespread accessibility, a web-based portal for submitting automatic segmentation jobs is currently being developed in collaboration with David Lee, Christopher Churas, and Willy Wong. The portal provides an intuitive and easy-to-use front

end for the methods described here; the user only has to select their training data, select their images to segment, and click the submit button. This eliminates the need for users to have to download, install, and compile software on their own, which should make the workflow easily accessible to a broad user base.

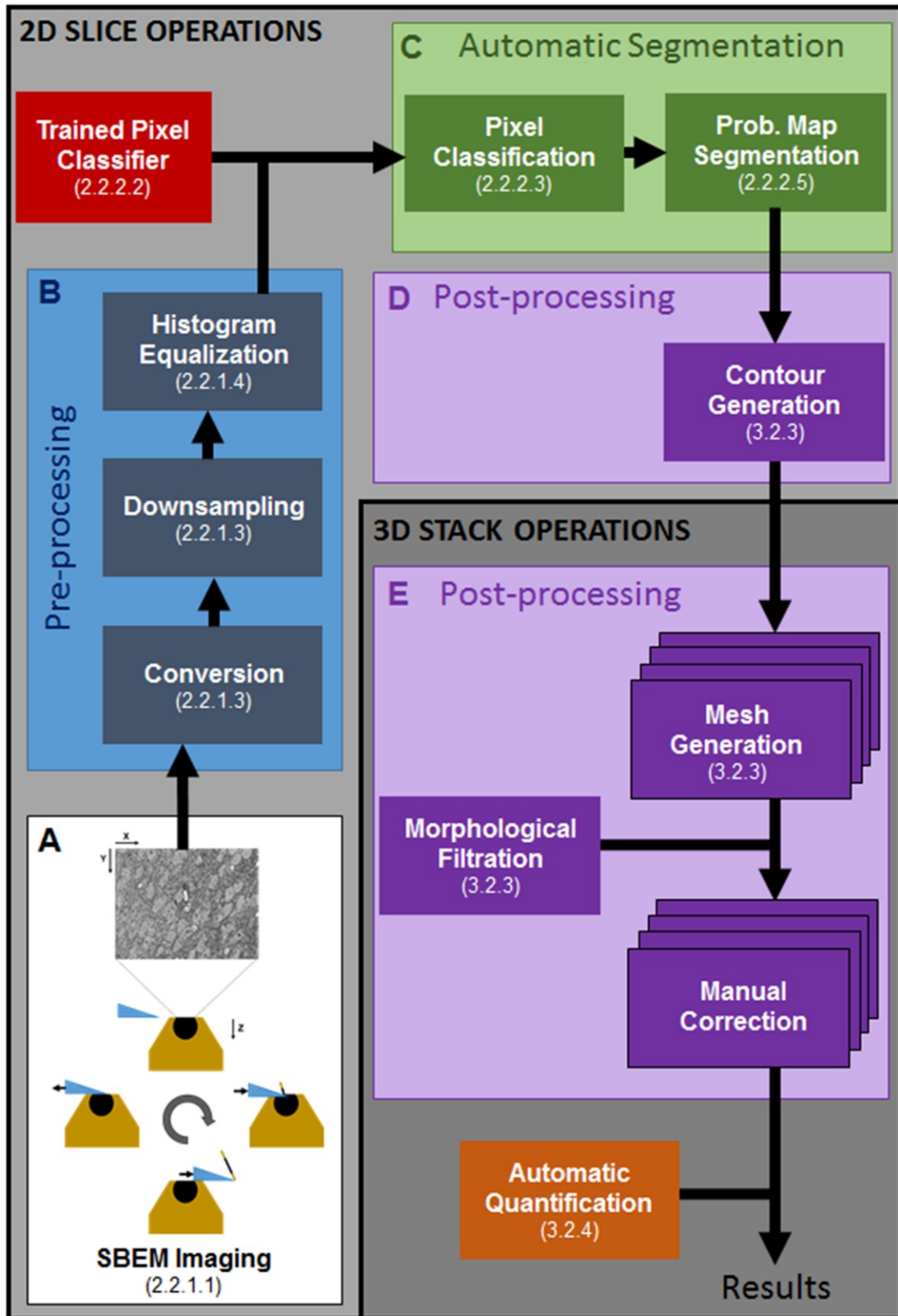
The techniques presented here are based upon the generation of 2D pixel classifications using the cascaded hierarchical model. However, the novel techniques described in Chapters 2 and 3 are all equally applicable to probabilities provided by any other 2D classification scheme. Thus, as the machine learning field evolves and better classification algorithms are developed, such approaches can be easily slotted into the proposed workflow. In addition to being more appropriate for anisotropic SBEM datasets, 2D pixel classifications are also more robust against common SBEM imaging defects such as focal gradients, obscured slices, and surface charging. Furthermore, 2D pixel classification facilitates a high degree of parallelization; in theory, all images could be simultaneously classified and segmented in parallel if one had access to enough computational resources to allow it. This also sets up the potential for images to be processed and segmented as they are collected, and this scenario will be explored in the future. A flowchart illustrating how the pipeline described in this dissertation would fit into such a scenario is shown in Figure 5.2. The same workflow with the interslice interpolation step added to it is shown in Figure 5.3. One drawback of such a design is that it relies upon a pre-trained classifier. Such a classifier could be supplied from a dataset that is similar in terms of appearance, pixel size, and feature of interest. Whether accurate segmentations can be obtained from a classifier trained against a different dataset remains to be determined, and this will be explored in the future. Alternatively, training data could be manually generated from the first few slices of the image stack after they have been acquired.

Workflows for the automatic quantification of cytoplasmic organelles are currently being developed. Such workflows will automatically provide morphological and spatial data in a manner analogous to the workflow presented for nuclei (Chapter 3.2.4.; Appendix D). A proof-of-concept illustration of this approach is shown in Figure 5.4 for mitochondrial characterization. Individual mitochondrial volumes were computed and displayed as a histogram, and the total mitochondrial volume and volume fraction were also calculated. As described previously, such analyses depend on accurate segmentations of cellular boundaries to group mitochondria into their appropriate neuron. In the preliminary data presented here, such boundaries were generated by manual segmentation (Figure 5.5). However, in the near future, methods for automatic segmentation of cell membranes will be explored and implemented into the workflow.

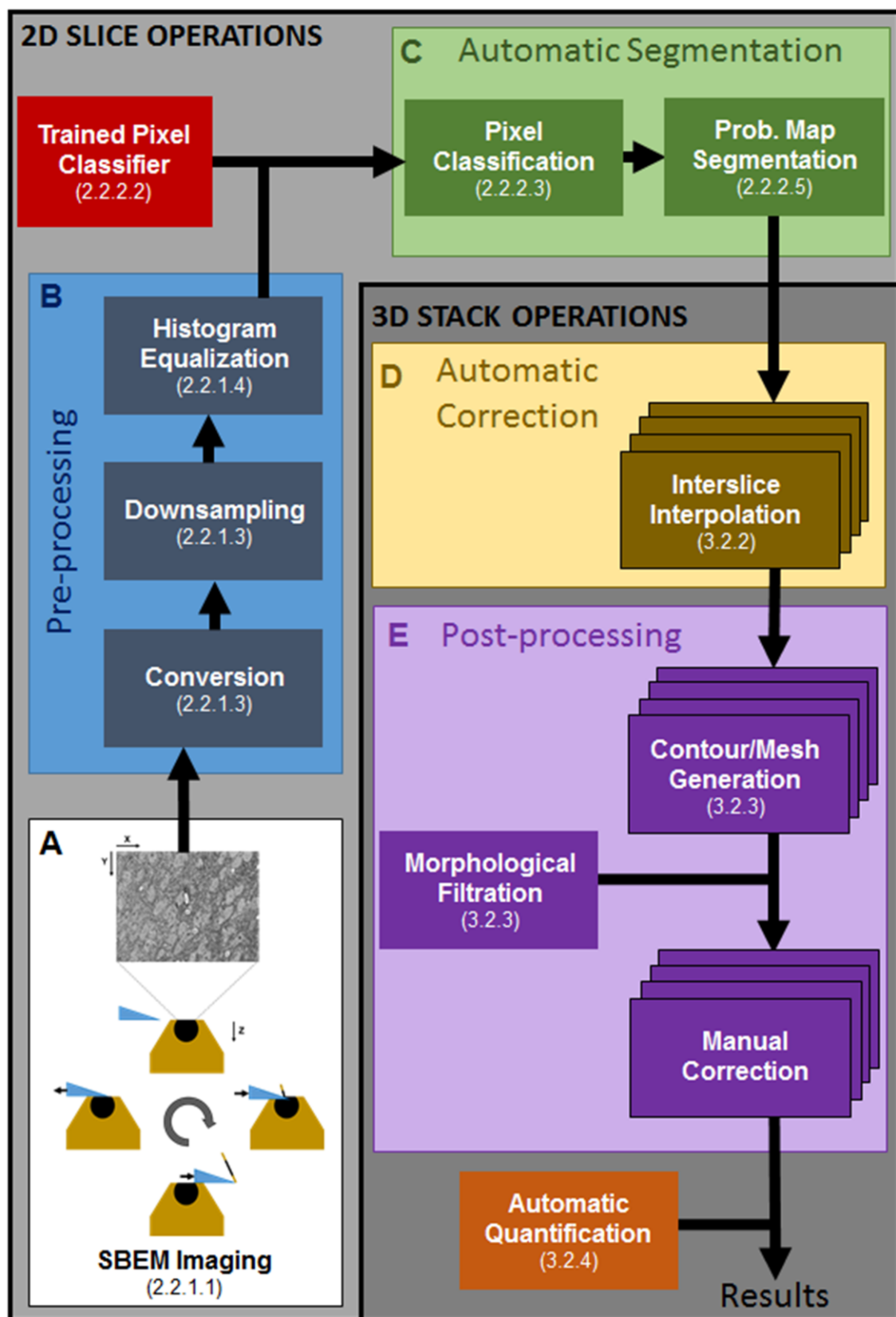
In summary, the work presented in this dissertation provides a novel method for the automatic segmentation of organelles and the quantification of their morphologies in 3D EM datasets. The validity of the workflow was tested and established by applying it to diverse organelle targets in numerous datasets of the mouse SCN. Two novel algorithms for increasing segmentation accuracy were developed and introduced. Numerous steps, including image segmentation and model generation, were expedited via parallelization on supercomputing resources. As the field continues to progress towards its goal of reconstructing entire nervous systems, these tools address a critical need by allowing for the quantitative analysis of volumetric EM datasets at a scale between that of current connectomics approaches (Kim, et al., 2014; Helmstaedter, et al., 2013) and that afforded by genetically encoded markers for small molecule localization (Martell, et al, 2012; Shu, et al., 2011).



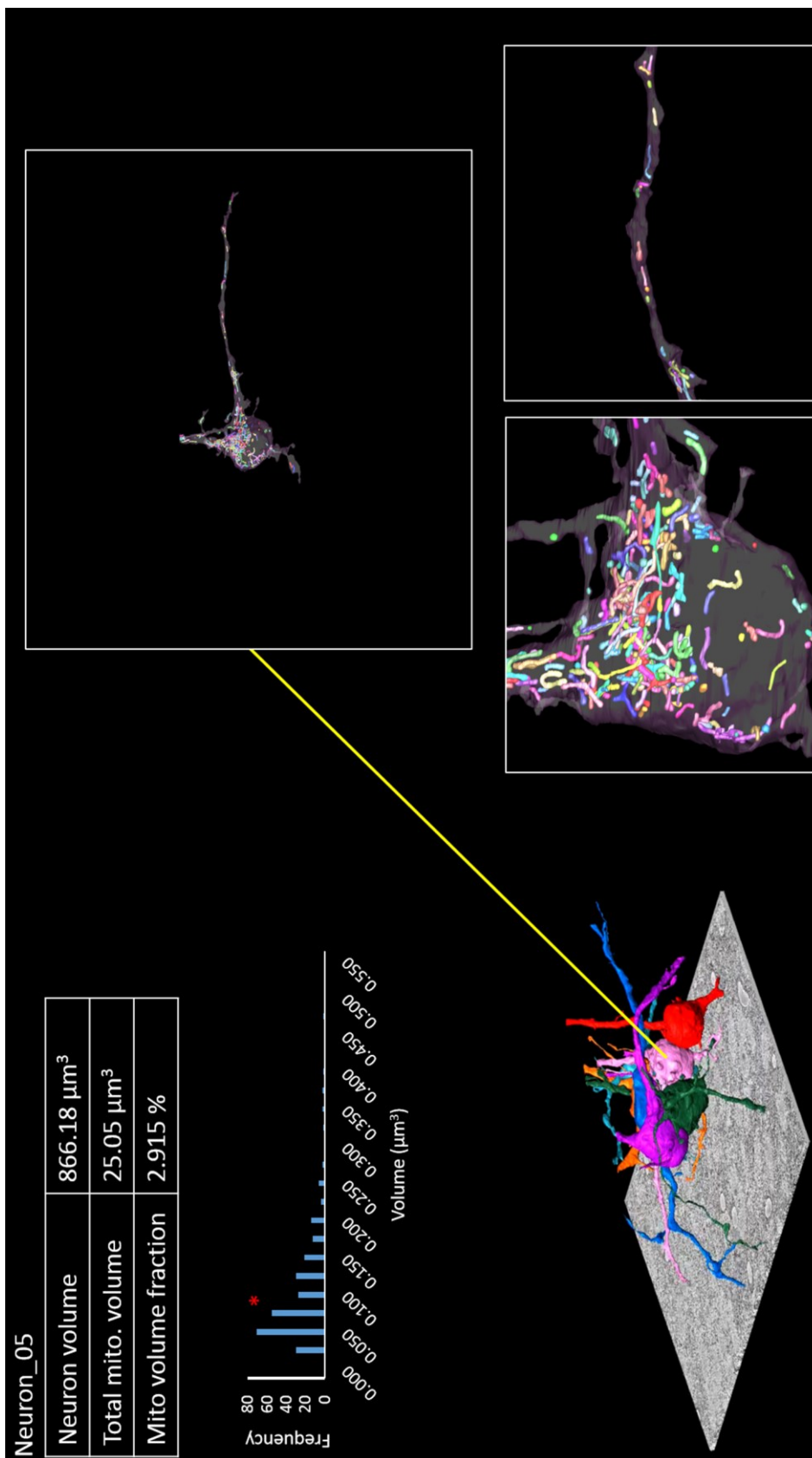
**Figure 5.2. A flowchart of the workflow demonstrating proposed future developments.** The flowchart begins in the bottom-left corner (A). The chapter in which each step was described is given in parentheses. Processing is initialized for every SBEM image after it has been acquired. Each image is subjected to conversion, downsampling, and histogram equalization using the expedited methods outlined in this dissertation (B). Next, the image is classified using a previously trained CHM pixel classifier (C), and the output probability map is binarized using the active contour evolution algorithm as previously described. As an alternative to using a previously trained classifier, the user may wait until a few slices have been collected, generate training data from them, and train a new, dataset-specific classifier. After probability map binarization, contours are generated around each 2D connected component (D). All steps through this point are performed in parallel for each newly acquired slice. After image acquisition has been terminated and contour models of the segmentation on each slice have been generated, the 3D stack operations are initiated. Meshes are generated (E), and morphological filters are applied to exclude erroneously large or small objects. Alternatively, an object classifier may be applied at this stage. The output model file is then manually inspected for accuracy and corrected as necessary. Once manual corrections have been completed, the automatic quantification workflow is initiated, and results are output to CSV files for further analyses.

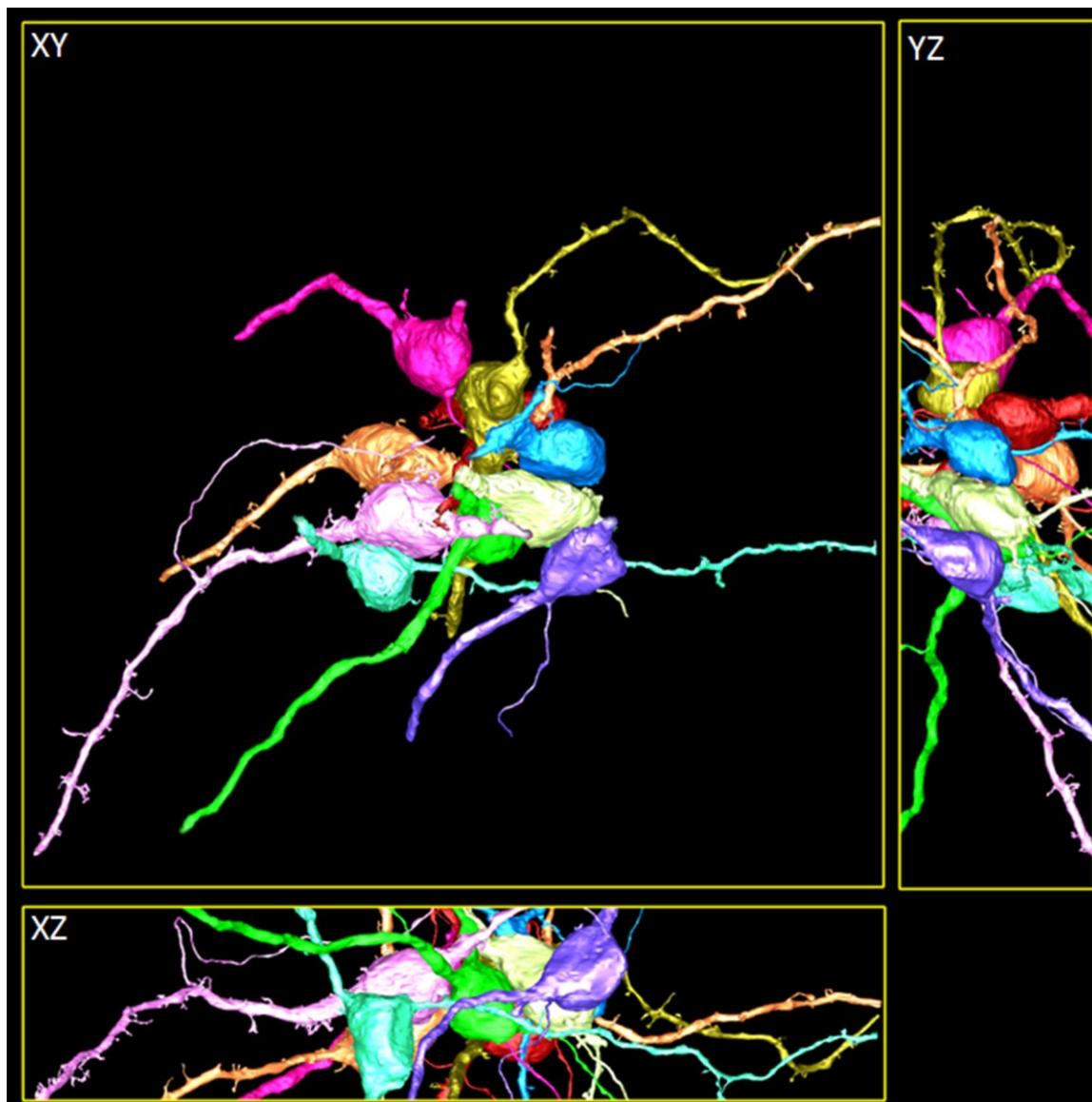


**Figure 5.3. A flowchart of the workflow demonstrating proposed future developments that includes automatic corrections.** Steps A-C are identical to those of the flowchart shown in Figure 5.2. However, in this scenario, the contour generation step cannot follow directly after segmentation since the entire 3D stack of segmentations is needed for interslice interpolation. Following the automatic correction of all objects, the workflow then proceeds the same as previously. The generation of contours and meshes is parallelized (E) using the methods described in Chapter 3.2.3.



**Figure 5.4. A demonstration of preliminary results from the automatic workflow for classifying mitochondrial morphology.** Mitochondria were automatically segmented and separated into their corresponding neuron using the methods described previously. All mitochondria in the neuron shown here were manually inspected and corrected as necessary. The bottom two panels demonstrate magnified views of parts of the neuronal soma (left) and axon (right). Mitochondria are preferentially localized to a specific side of the soma and continue down the entirety of the axon. The neuronal volume, total mitochondrial volume, and mitochondrial volume fraction were automatically calculated and are reported here. A histogram of individual mitochondrial volumes is also displayed. These data will be automatically output once the workflow has been finalized.





**Figure 5.5. A subset of ten manually segmented neurons.** Neurons were segmented and rendered using the methods previously described. Shown here are ten neurons from the second SCN ZT4-6 dataset. Both spiny (pink) and smooth (cyan) neurons are present in close proximity to one another. These segmentations will be used for the future analysis of cytoplasmic organelles.

## **Appendix A. A Survey of the Quantitative Methods Used in Published SBEM Studies**

The results of a literature survey of studies employing SBEM from 2006 through March 2014 are presented here. The citation for each study is given, along with a description of what was quantified, how this quantification was performed, and whether manual or semi-automatic segmentation and quantification methods were employed.



Citation	Manual or semi-automatic?	What was quantified?	How was the image analysis performed?
Feierbach et al., 2006	Manual	Length and width of nuclear filaments. Geometry of association between filaments and capsids.	Manual segmentation.
O'Connell et al., 2008	Semi-automatic	Nuclear and cytoplasmic volume fractions. Nuclear ellipticity and radial tilt. Volume, thickness, and density of radial elastin struts.	Combination of manual and semi-automatic segmentation (methods are unclear).
Armer et al., 2009	Semi-automatic	Diameter of dorsal lateral anastomotic vessels.	Fitting of Amira CurvedSlice projections to vessels.
Rouquette et al., 2009	Semi-automatic	Volume fractions of chromatin and interchromatin space.	Manual segmentation of nuclei, followed by the automatic segmentation of chromatin by single-level grayscale thresholding.
Pellettieri et al., 2010	Manual	Number of cells exhibiting chromatin condensation.	Manual counting.
West et al., 2010	Manual	No quantitative SBEM data reported.	Manual segmentation of capillaries and epithelial bridges.
Briggman et al., 2011	Manual	Distribution of angles between starburst amacrine cells (SAC) somata and synapses. Synapse locations relative to SAC somata.	Manual tracing of skeletons through dendritic trees. Tracing of skeletons from putative synapses back to their somata.
Kalson et al., 2011	Manual	Nuclear length and volume. Cross-sectional areas of nuclei and cells.	Manual segmentation of nuclei and cells.
Motskin et al., 2011	Manual	Number and size of vacuoles and SCC.	Manual segmentation of vacuoles and SCC.

Citation	Manual or semi-automatic?	What was quantified?	How was the image analysis performed?
Mun et al., 2011	Manual	Melanocyte volume and the distribution of melanocyte dendrites	Manual segmentation of melanocytes.
Mustafi et al., 2011	Manual	No quantitative SBEM data reported.	Manual segmentation of phagosomes, nuclei, and cells for visualization only.
Nguyen et al., 2011	Manual	Frequency and radius of granule accumulations.	Manual counting.
Powner et al., 2011	Manual	Degree of basement membrane vacuolization.	Manual segmentation.
Rezakhaniha et al., 2011	N/A	No quantitative SBEM data reported.	N/A
Shu et al., 2011	Semi-automatic	No quantitative SBEM data reported.	Maximum intensity projections for visualization only.
Ewald et al., 2012	Manual	No quantitative SBEM data reported.	Manual segmentation of membranes for visualization only.
Hatori et al., 2012	Manual	Volume fractions of various organelles.	Stereology using the IMOD plugin.
Hoppa et al., 2012	Manual	Percentage of fused granules and distances from granules to the plasma membrane	Not reported.
Luhmann et al., 2012	Manual	No quantitative SBEM data reported.	Manual segmentation of the vascular lumen for visualization only.

Citation	Manual or semi-automatic?	What was quantified?	How was the image analysis performed?
Ou et al., 2012	Manual	No quantitative SBEM data reported.	Manual segmentation of the nucleus, nucleolus, and plasma membrane for visualization only.
Pinheiro et al., 2012	Manual	No quantitative SBEM data reported.	Manual segmentation of nuclei.
Puhka et al., 2012	Semi-automatic	No quantitative SBEM data reported.	Processing performed using the Microscopy Image Browser.
Schwartz et al., 2012	Manual	No quantitative SBEM data reported.	Manual segmentation of a whole cell for visualization only.
Zhuraleva et al., 2012	Manual	Mitochondrial volume and length.	Manual segmentation of mitochondria and their approximation as ellipsoids.
Anttonen et al., 2013	Manual	Length of junctions between Deiters' cells.	Distance measurements in Amira.
Boassa et al., 2013	Manual	No quantitative SBEM data reported.	Manual segmentation of membranes for visualization only.
Chung et al., 2013	Manual	No quantitative SBEM data reported.	Manual segmentation of membranes for visualization only.
Helmstaedter et al., 2013	Semi-automatic	Skeleton density vs. tissue depth. Frequencies of synaptic and non-synaptic contacts. Cell-cell contact matrices.	Manual, crowd-sourced tracing of dendritic skeletons. Manual detection of cell type by student segmenters. Automatic segmentation of cell boundaries using a convolutional network. Combination of skeletons with segmentations to determine connectedness.

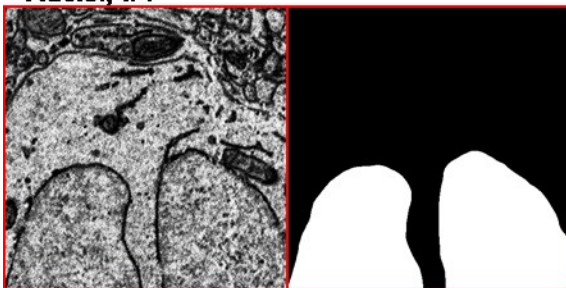
Citation	Manual or semi-automatic?	What was quantified?	How was the image analysis performed?
Hermes et al., 2013	Manual	Volume fraction of lipid droplets in hepatocytes.	Manual segmentation of lipid droplets, nuclei, and plasma membranes.
Holcomb et al., 2013	Manual	Number of inputs versus apposed surface area.	Manual segmentation of nuclei and plasma membranes.
Hu et al., 2013	N/A	No quantitative SBEM data reported.	N/A
Hughes et al., 2013	Manual	Flagellar length and position within the cell.	Manual segmentation and length measurements in Amira
Kalson et al., 2013	Manual	Collagen fibril length.	Manual segmentation and length measurements of collagen fibrils. Manual segmentation of the plasma membrane.
Keeley et al., 2013	Manual	Number of mitochondria and invaginating elements per pedicle.	Manual segmentation of the plasma membrane and mitochondria.
Lafontant et al., 2013	Manual	No quantitative SBEM data reported.	Manual segmentation of membranes.
Mustafi et al., 2013	Manual	Number of phagosomal events per RPE cell.	Manual segmentation of membranes and phagosomes.
Najm et al., 2013	Manual	No quantitative SBEM data reported.	Manual segmentation of membranes.
Peridaen et al., 2013	Manual	Percentage of cells displaying certain membrane structures about primary cilia.	Inspection and counting of cell types.

Citation	Manual or semi-automatic?	What was quantified?	How was the image analysis performed?
Pinali et al., 2013	Semi-automatic	Morphological parameters of t-tubules and the jSR-sarcolemma region.	“Contrast-based” segmentation.
Pollier et al., 2013	Semi-automatic	No quantitative SBEM data reported.	Seeded watershed using ilastik.
Wilke et al., 2013	Manual	Morphological characterization of dendritic spines. Volume and surface area. Mitochondrial volume.	Manual segmentation of plasma membrane, mitochondria, and organelles.
Wong et al., 2013	Manual	No quantitative SBEM data reported.	Manual segmentation of membranes.
Xiao et al., 2013	Manual	Number and volume of nerve terminals.	Manual segmentation of membranes and synapses.
Arkill et al., 2014	Manual	Length and width of sub-podocyte spaces.	Manual segmentation of membranes.
Bohorquez et al., 2014	Manual	Number and distribution of secretory vesicles.	Manual segmentation of nuclei and membranes. Localization of secretory vesicles.
Lipke et al., 2014	Manual	Volume of sperm cells.	Manual segmentation of membranes.
Pingel et al., 2014	Manual	Distribution of collagen fibril diameters.	Manual segmentation of fibrils.
Wilke et al., 2014	Manual	Bouton and spine volumes. Area of synaptic contacts and vesicle numbers.	Manual segmentation of membranes and mitochondria.
Young et al., 2014	Manual	Orientation of collagen fibrils with respect to one another using a Fourier analysis.	Manual segmentation of collagen bundles.

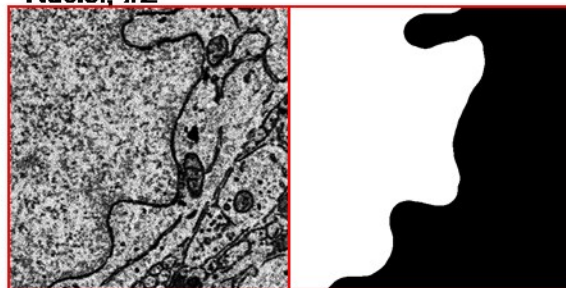
## **Appendix B. Training Images and Labels**

The training images and labels used in Chapter 2 for the generation of organelle-specific CHM probability maps are displayed on the following pages. All training images were of the size 500 x 500 pixels and were extracted from the original stack after it was downsampled by a factor of two. Training labels were generated by manual segmentation of all organelles of interest using IMOD.

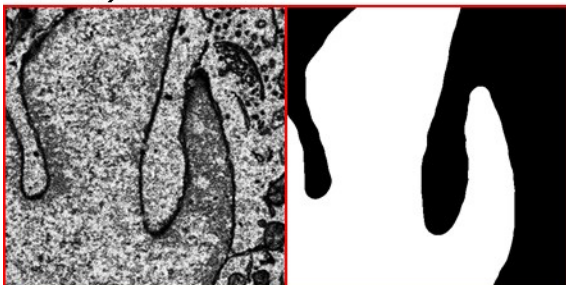
Nuclei, #1



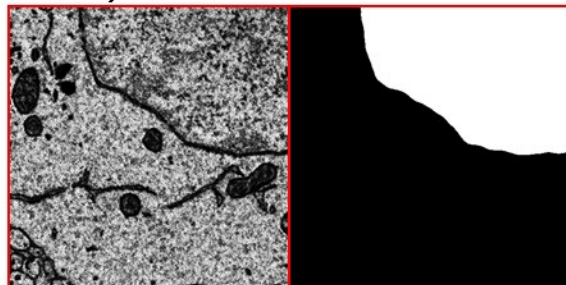
Nuclei, #2



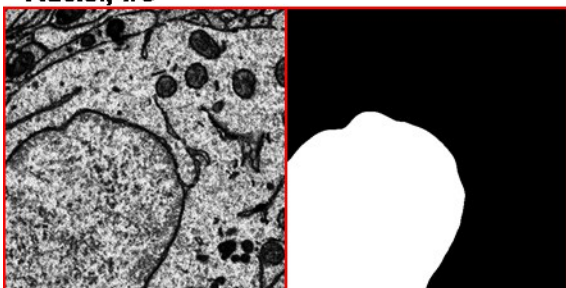
Nuclei, #3



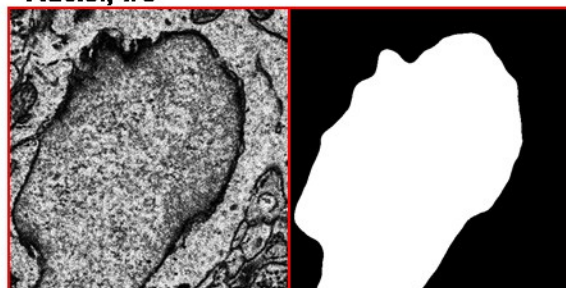
Nuclei, #4



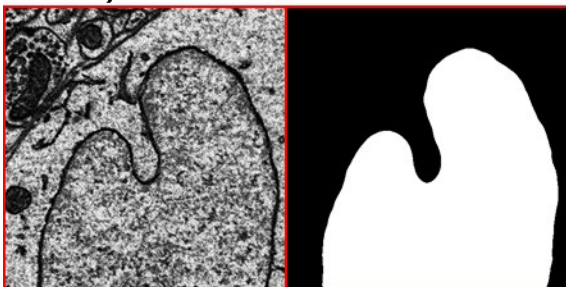
Nuclei, #5



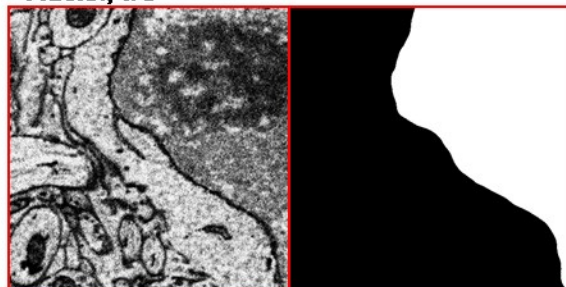
Nuclei, #6



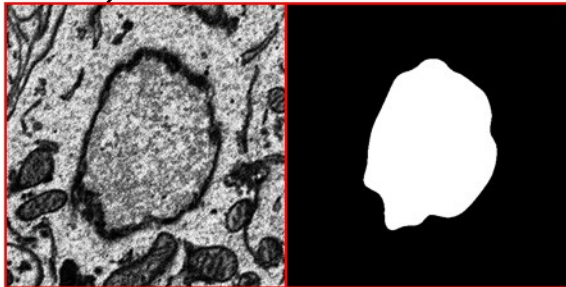
Nuclei, #7



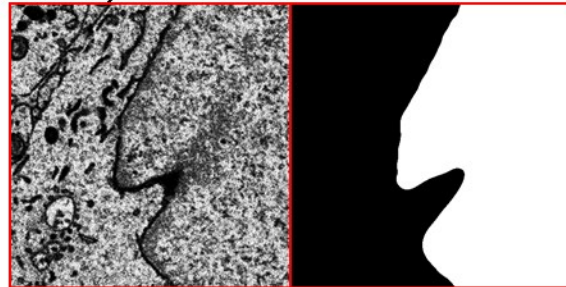
Nuclei, #8



Nuclei, #9

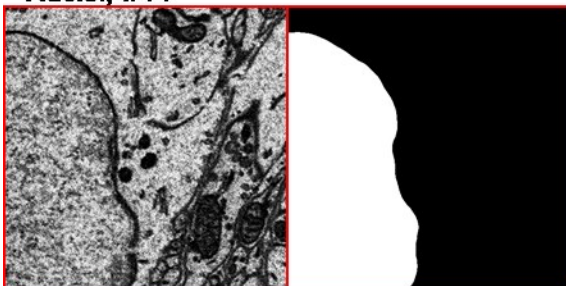


Nuclei, #10

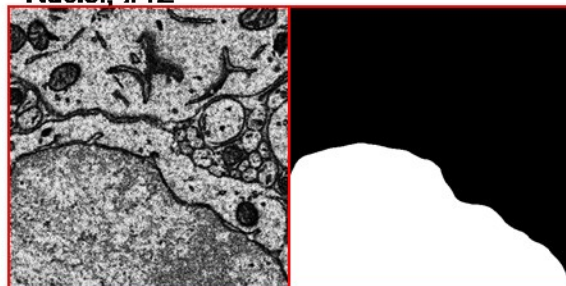




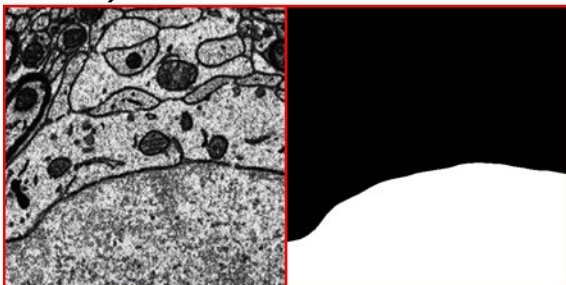
Nuclei, #11



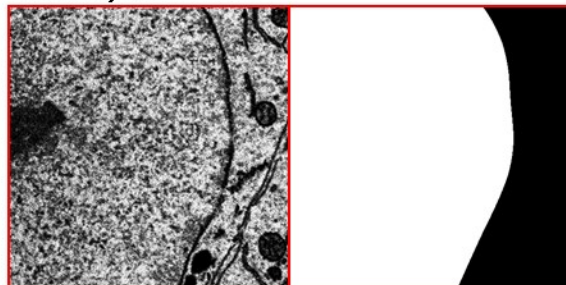
Nuclei, #12



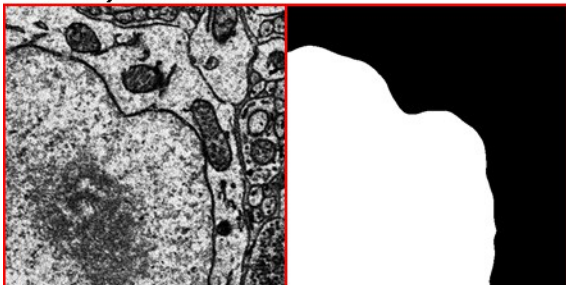
Nuclei, #13



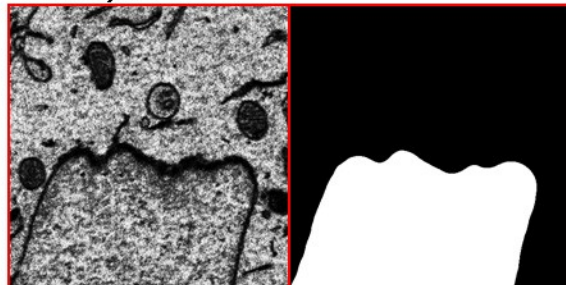
Nuclei, #14



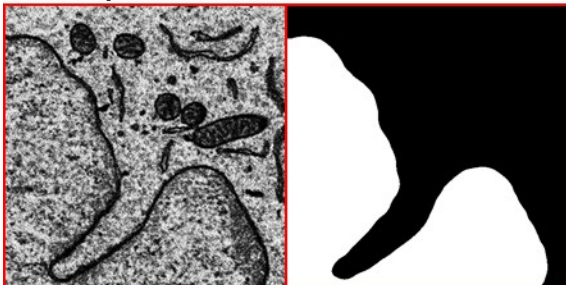
Nuclei, #15



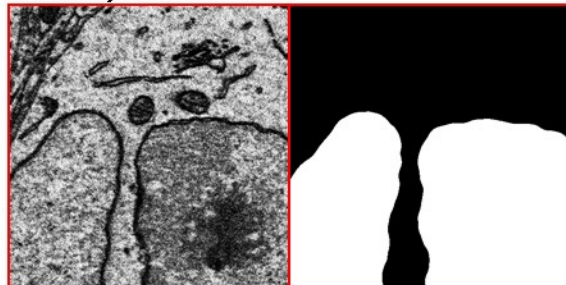
Nuclei, #16



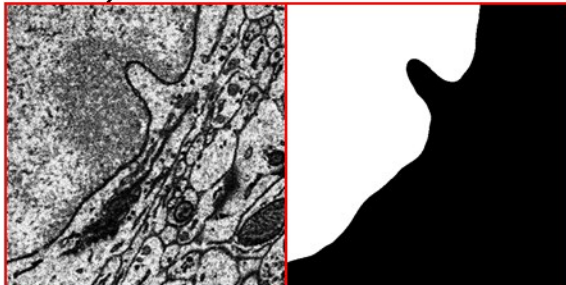
Nuclei, #17



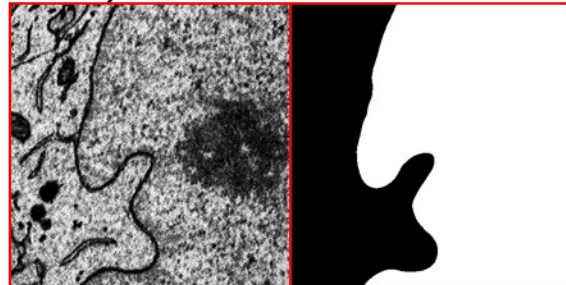
Nuclei, #18



Nuclei, #19

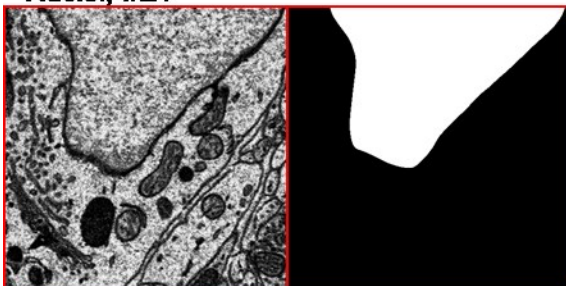


Nuclei, #20





Nuclei, #21



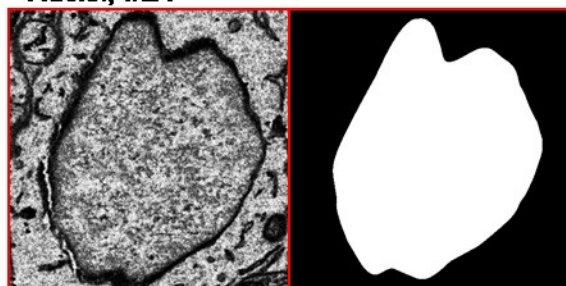
Nuclei, #22



Nuclei, #23



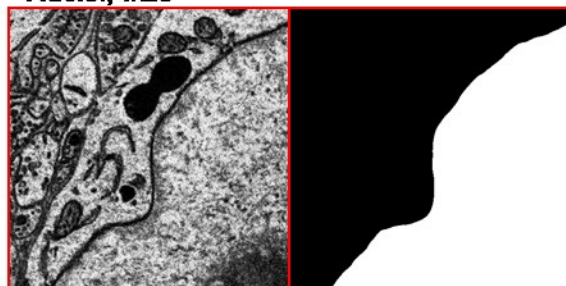
Nuclei, #24



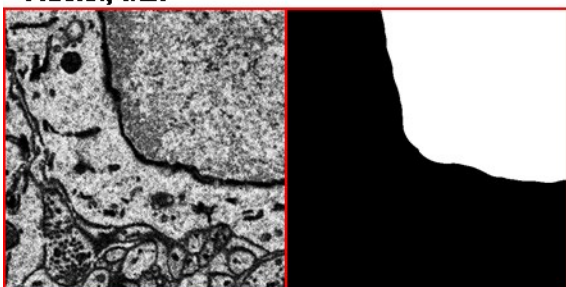
Nuclei, #25



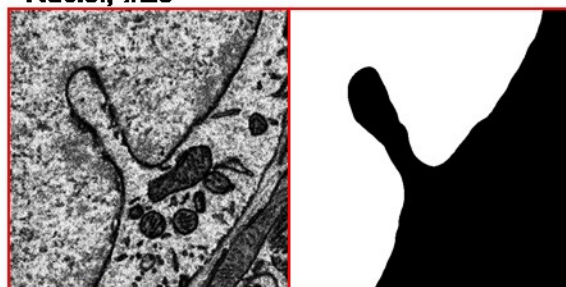
Nuclei, #26



Nuclei, #27



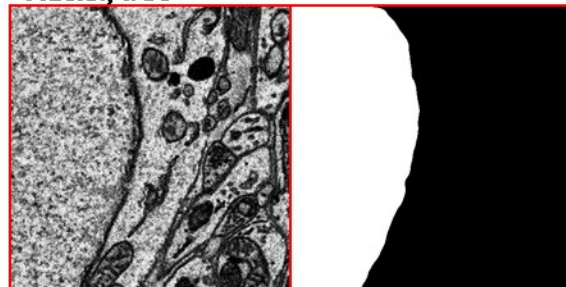
Nuclei, #28



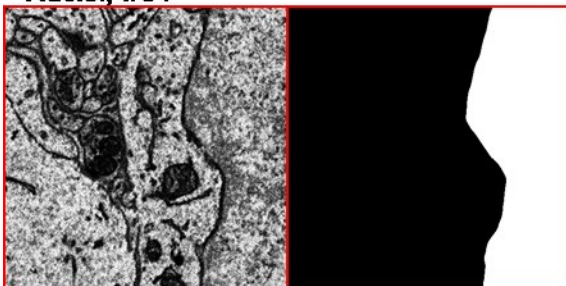
Nuclei, #29



Nuclei, #30



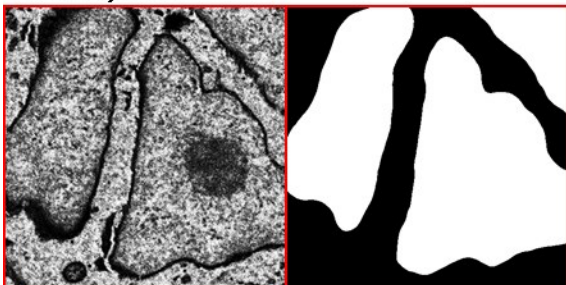
Nuclei, #31



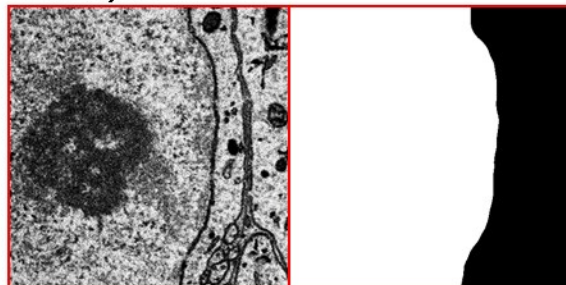
Nuclei, #32



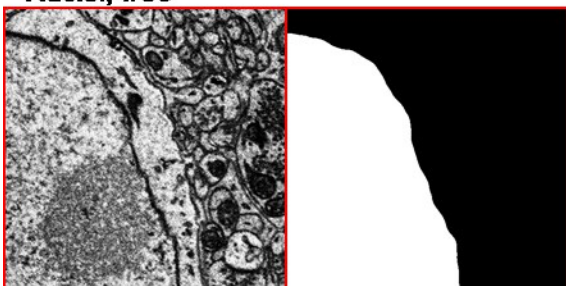
Nuclei, #33



Nuclei, #34



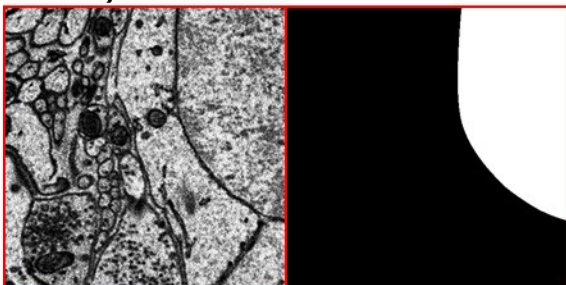
Nuclei, #35



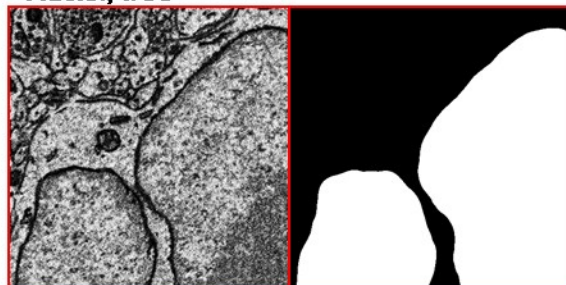
Nuclei, #36



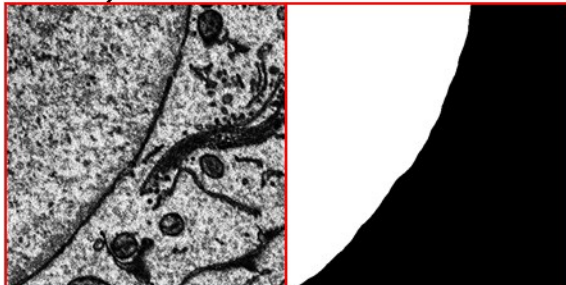
Nuclei, #37



Nuclei, #38



Nuclei, #39

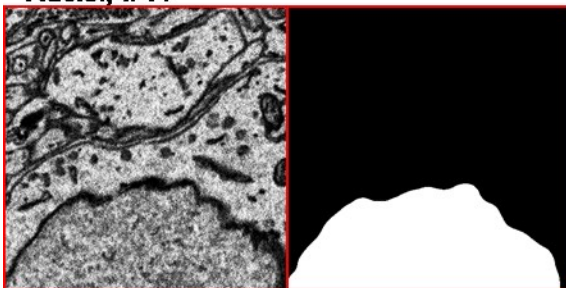


Nuclei, #40

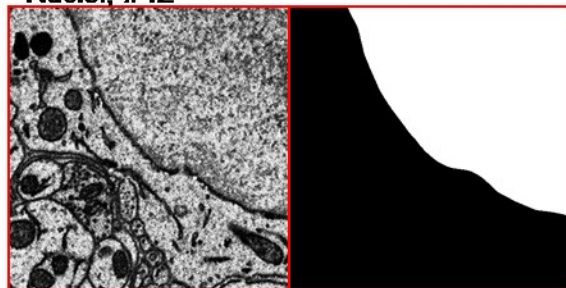




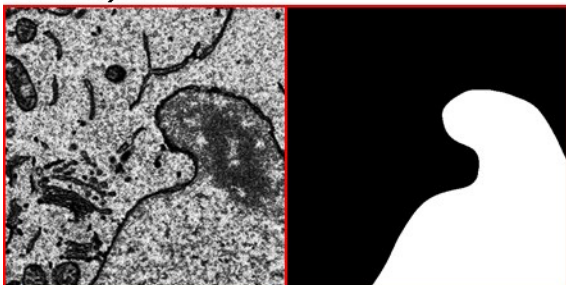
Nuclei, #41



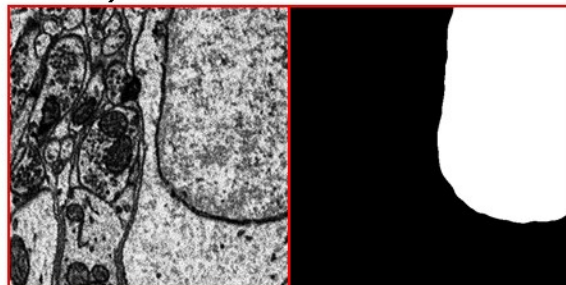
Nuclei, #42



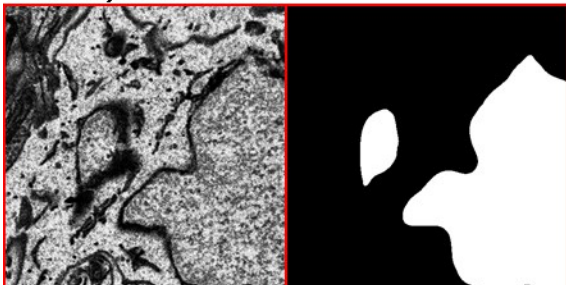
Nuclei, #43



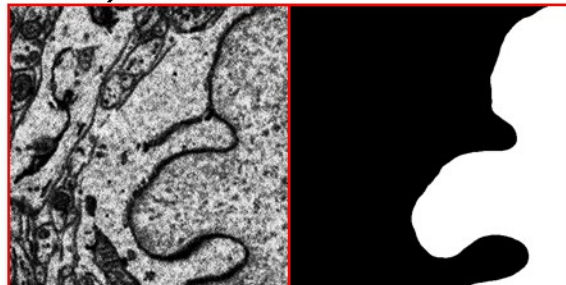
Nuclei, #44



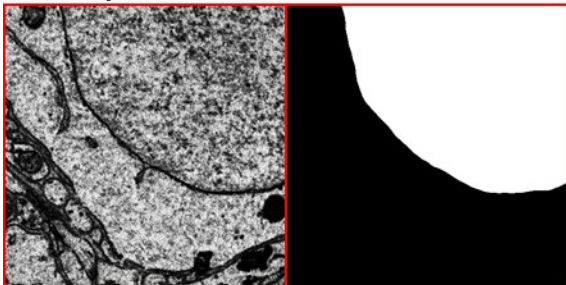
Nuclei, #45



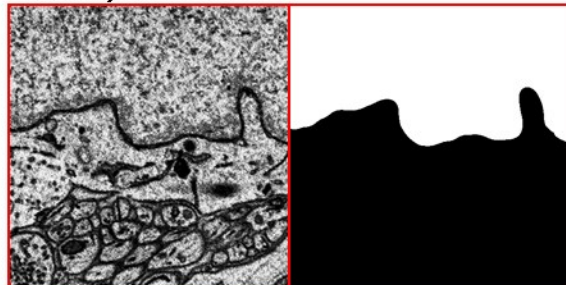
Nuclei, #46



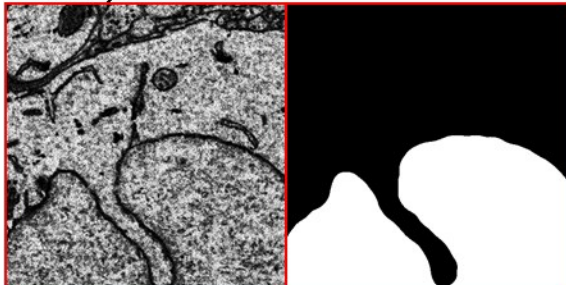
Nuclei, #47



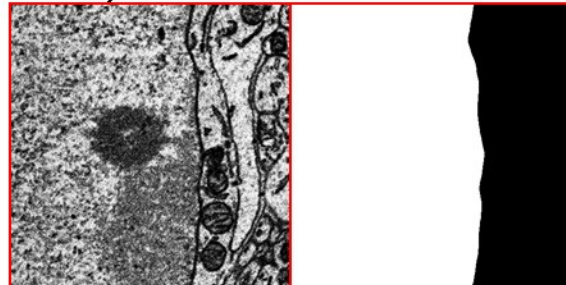
Nuclei, #48



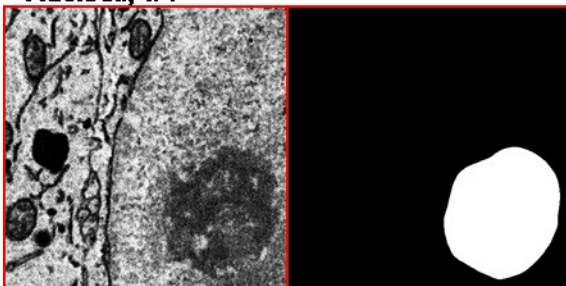
Nuclei, #49



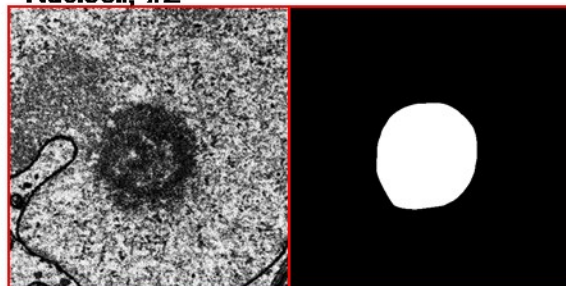
Nuclei, #50



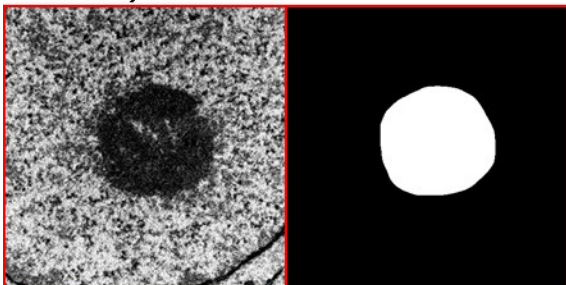
Nucleoli, #1



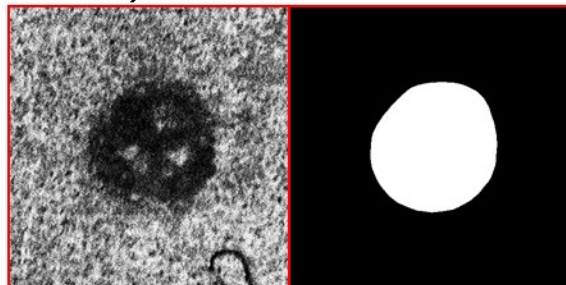
Nucleoli, #2



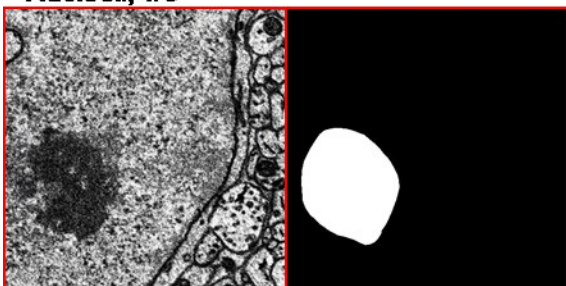
Nucleoli, #3



Nucleoli, #4



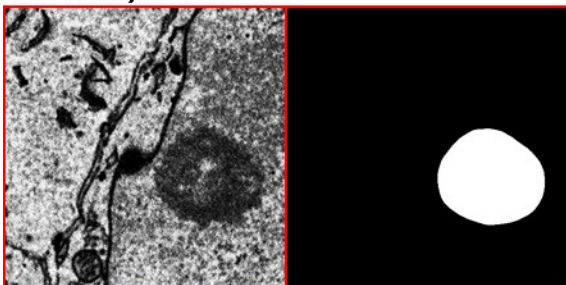
Nucleoli, #5



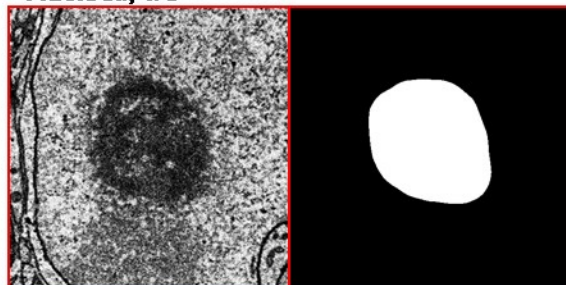
Nucleoli, #6



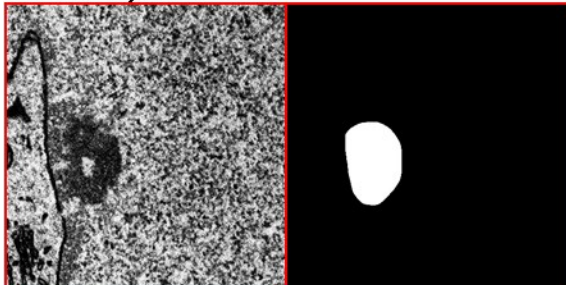
Nucleoli, #7



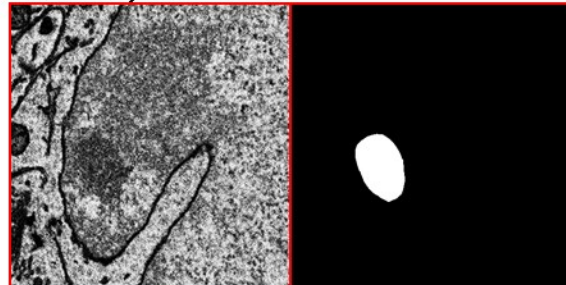
Nucleoli, #8



Nucleoli, #9

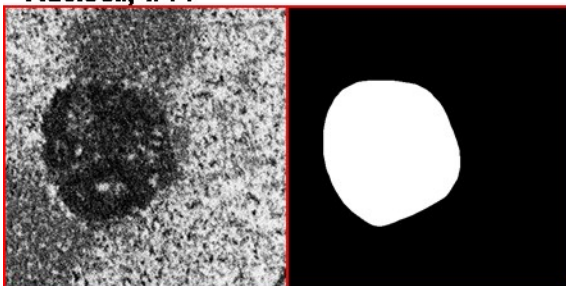


Nucleoli, #10

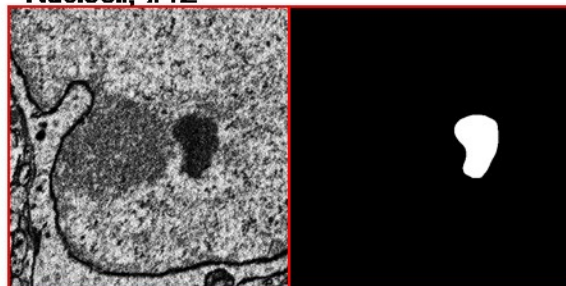




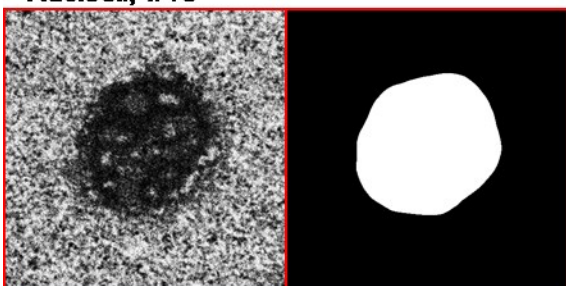
Nucleoli, #11



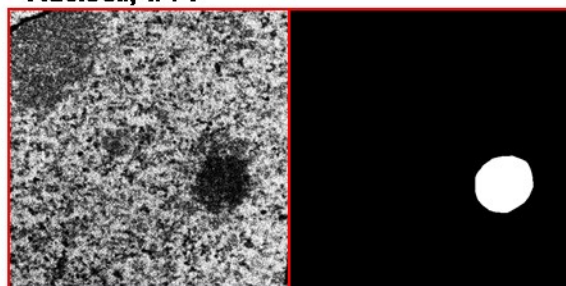
Nucleoli, #12



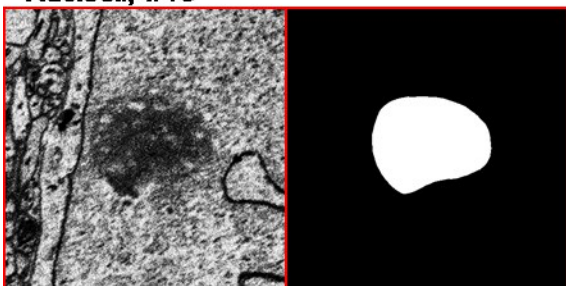
Nucleoli, #13



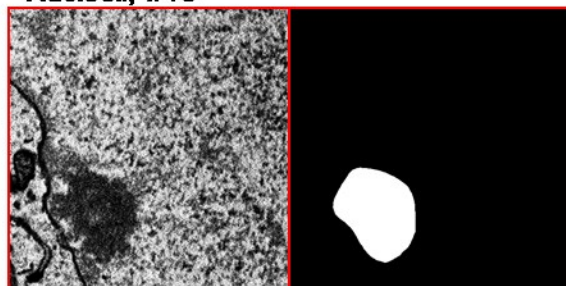
Nucleoli, #14



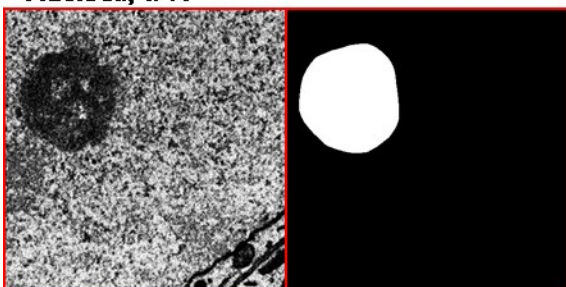
Nucleoli, #15



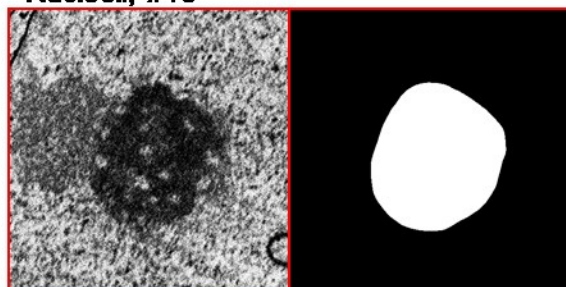
Nucleoli, #16



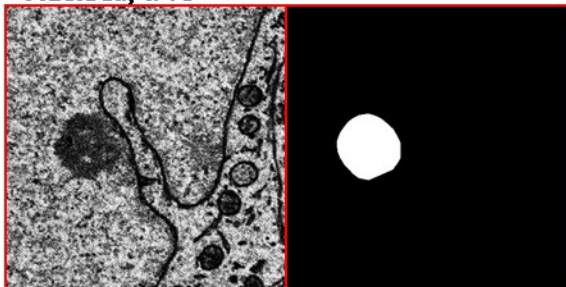
Nucleoli, #17



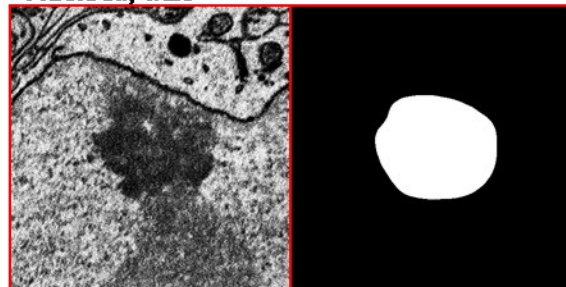
Nucleoli, #18



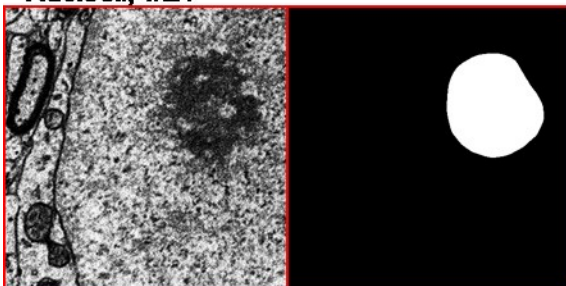
Nucleoli, #19



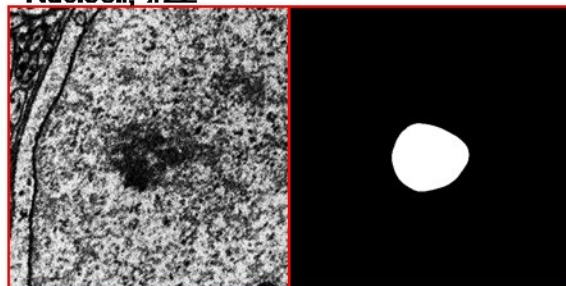
Nucleoli, #20



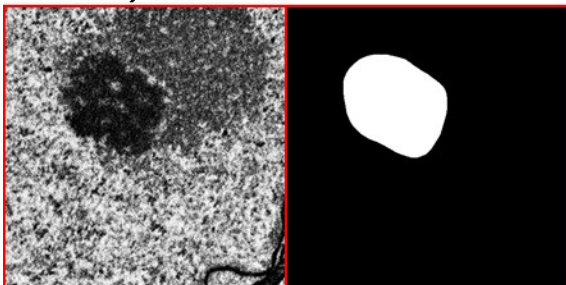
Nucleoli, #21



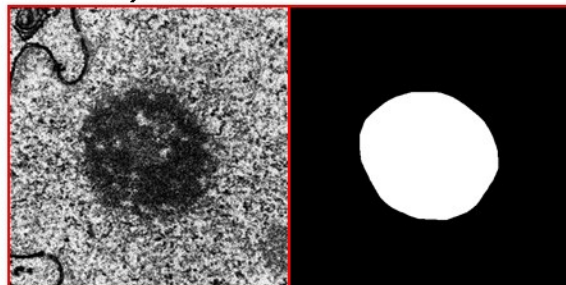
Nucleoli, #22



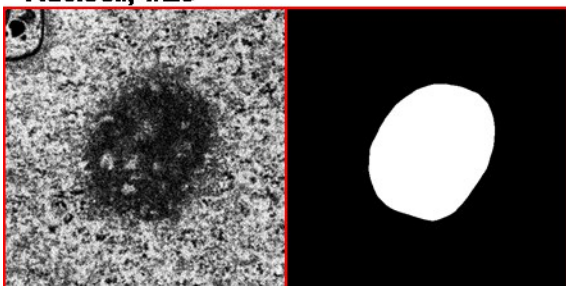
Nucleoli, #23



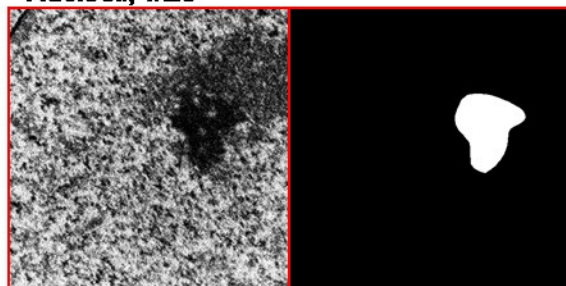
Nucleoli, #24



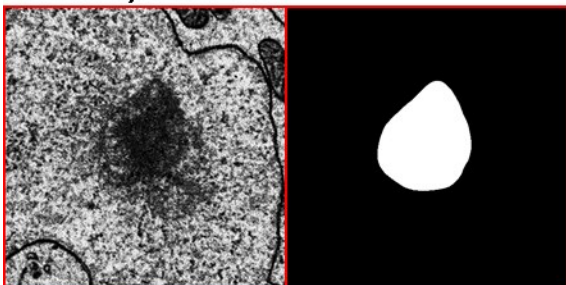
Nucleoli, #25



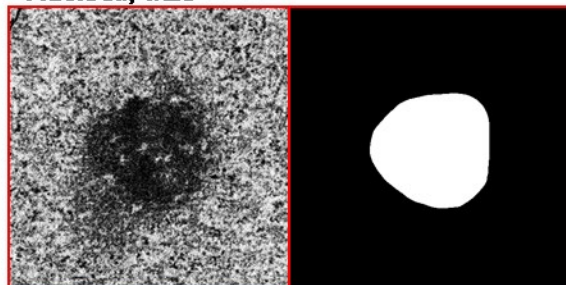
Nucleoli, #26



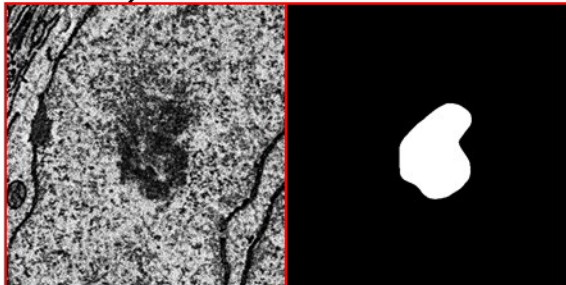
Nucleoli, #27



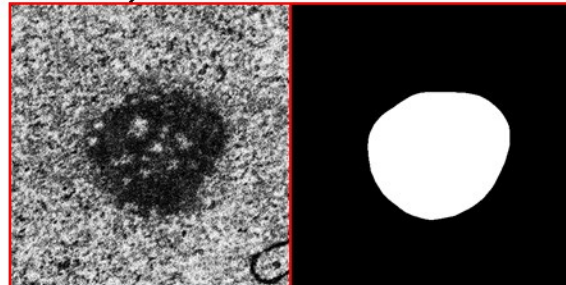
Nucleoli, #28



Nucleoli, #29

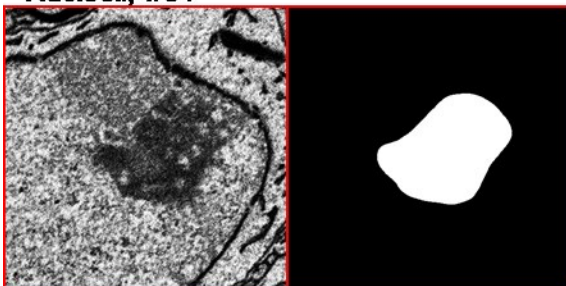


Nucleoli, #30

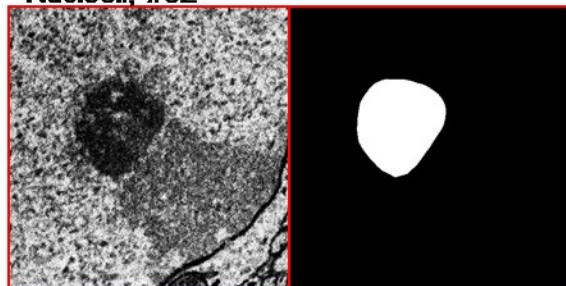




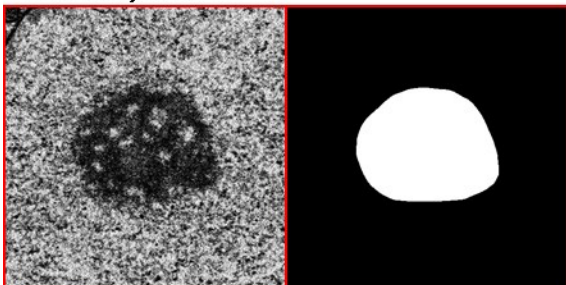
Nucleoli, #31



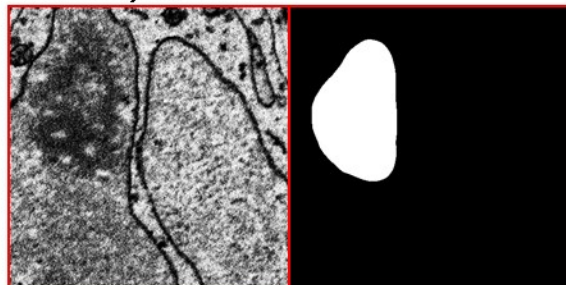
Nucleoli, #32



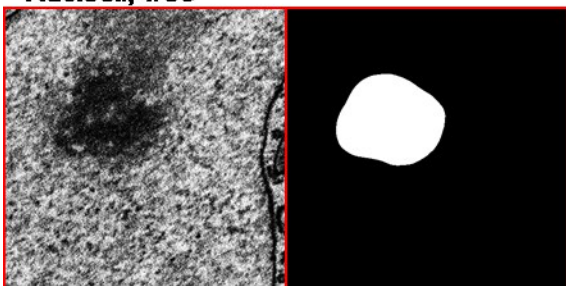
Nucleoli, #33



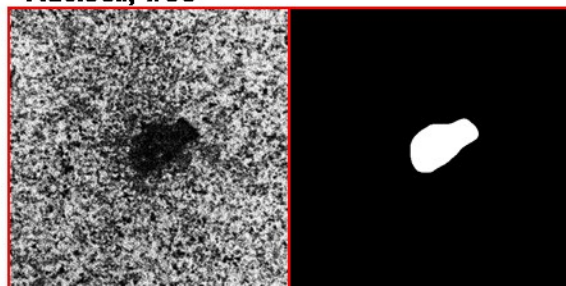
Nucleoli, #34



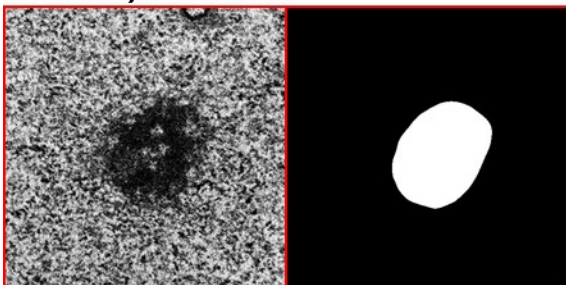
Nucleoli, #35



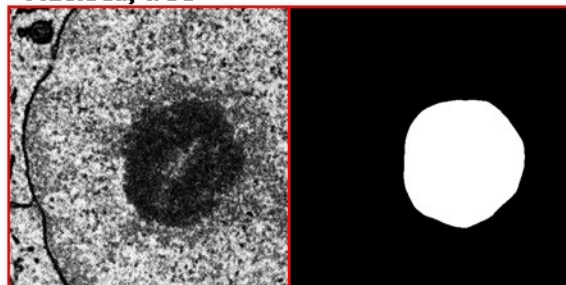
Nucleoli, #36



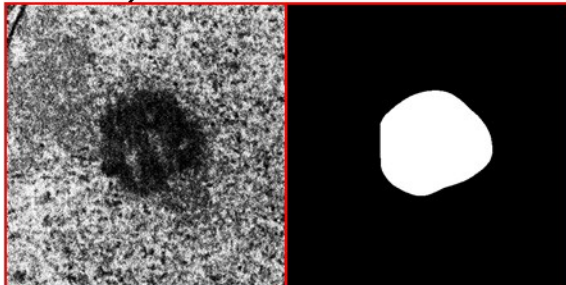
Nucleoli, #37



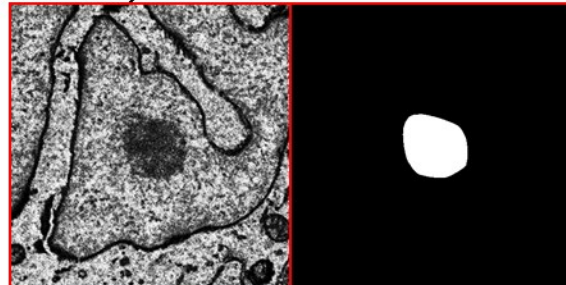
Nucleoli, #38



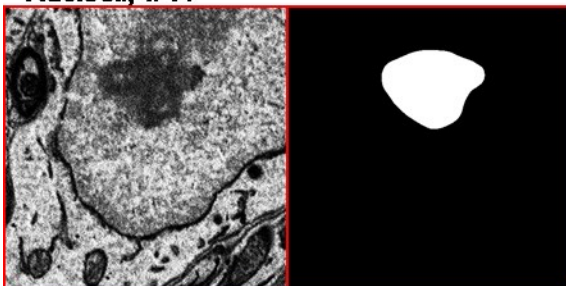
Nucleoli, #39



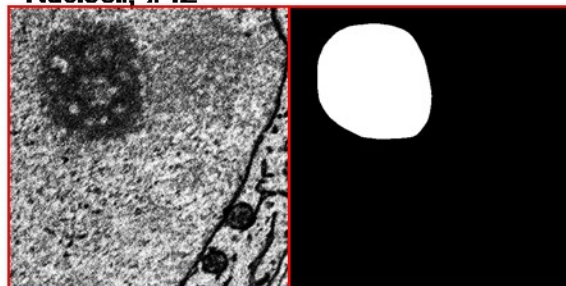
Nucleoli, #40



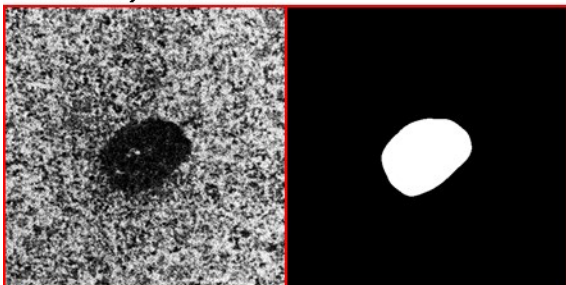
Nucleoli, #41



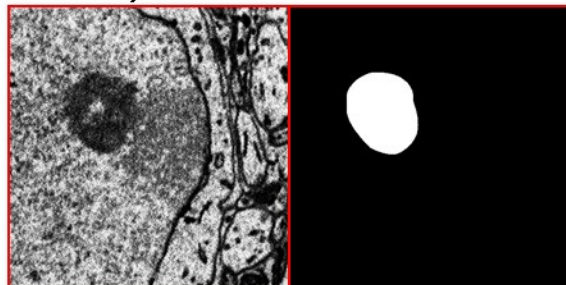
Nucleoli, #42



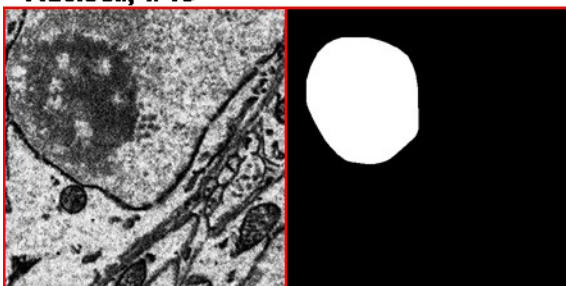
Nucleoli, #43



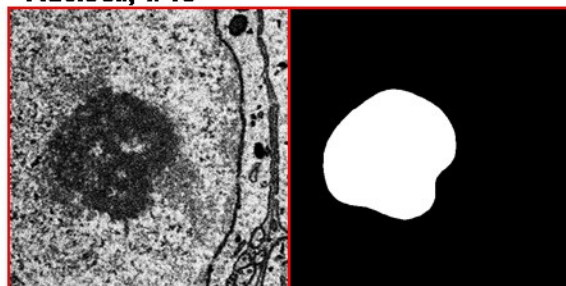
Nucleoli, #44



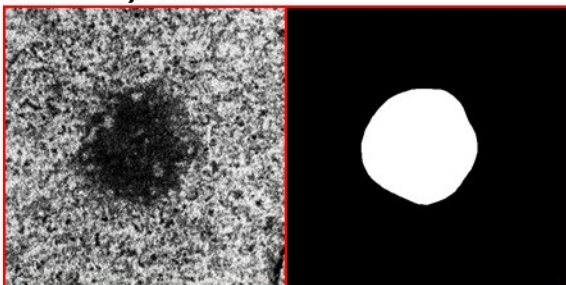
Nucleoli, #45



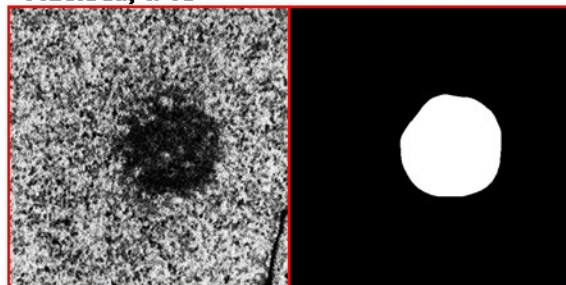
Nucleoli, #46



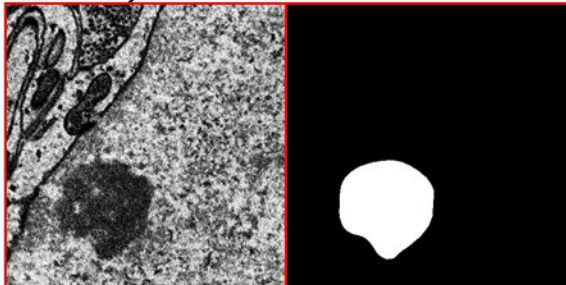
Nucleoli, #47



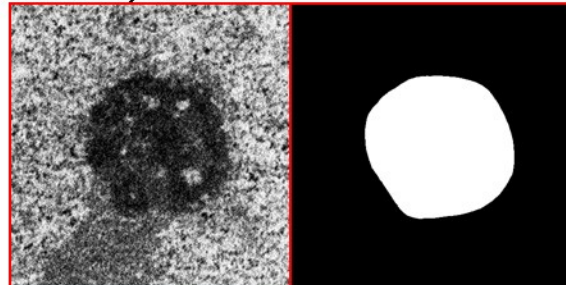
Nucleoli, #48



Nucleoli, #49

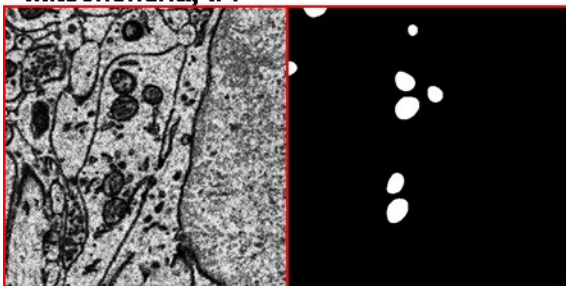


Nucleoli, #50





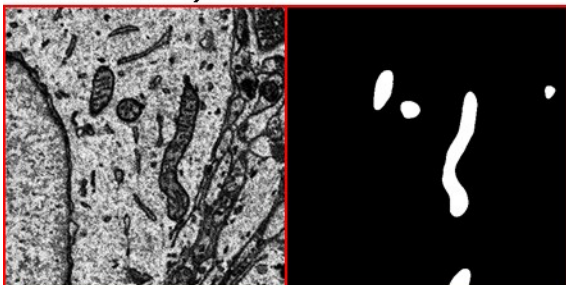
Mitochondria, #1



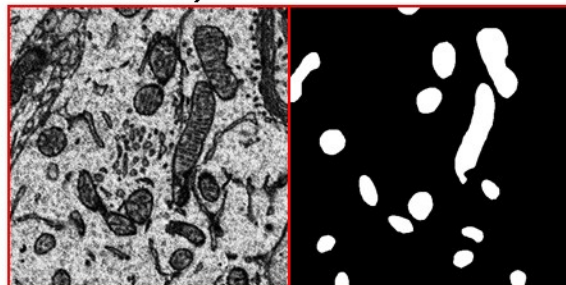
Mitochondria, #2



Mitochondria, #3



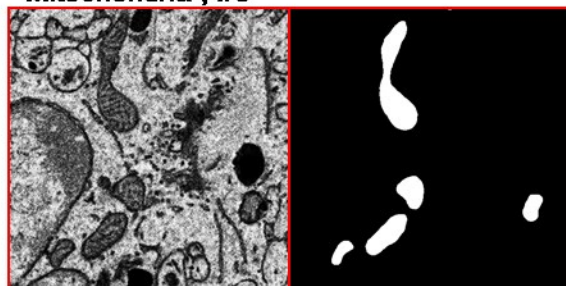
Mitochondria, #4



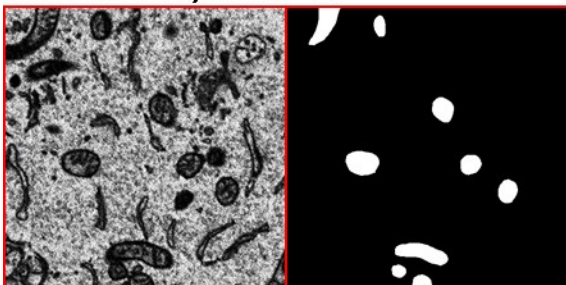
Mitochondria, #5



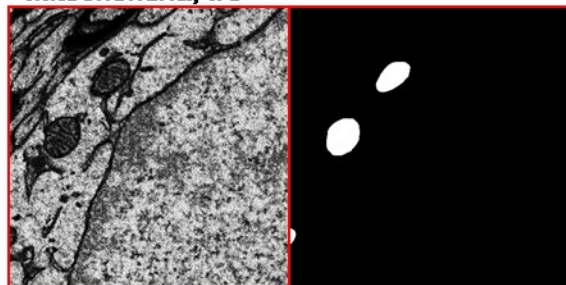
Mitochondria, #6



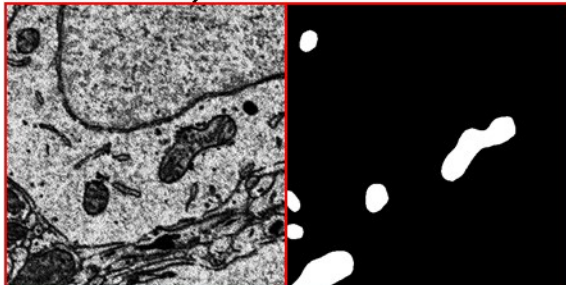
Mitochondria, #7



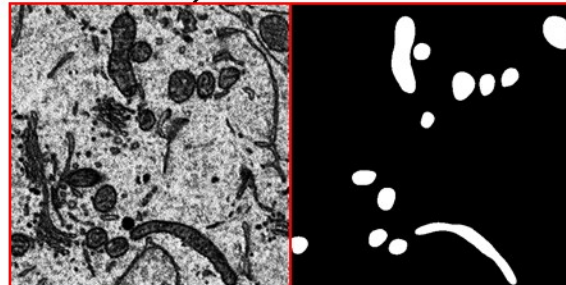
Mitochondria, #8



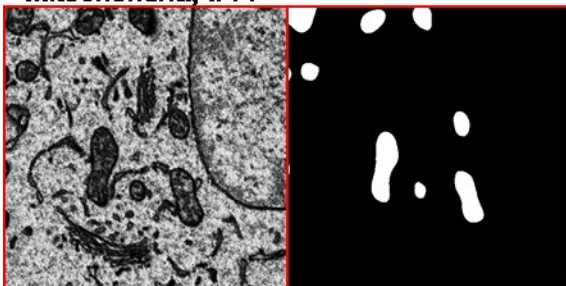
Mitochondria, #9



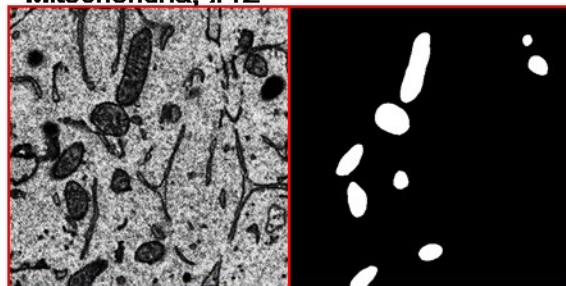
Mitochondria, #10



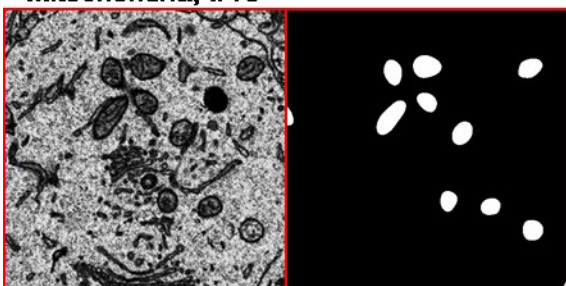
Mitochondria, #11



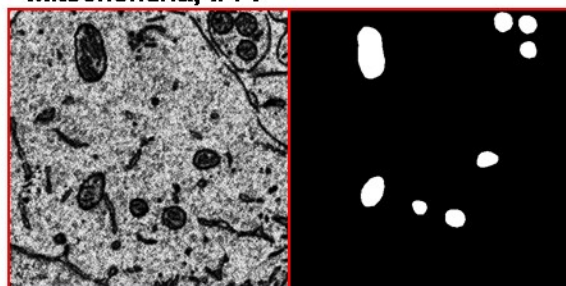
Mitochondria, #12



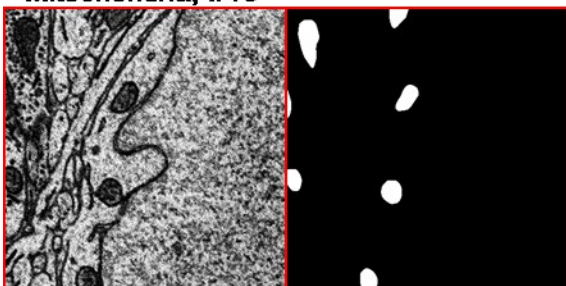
Mitochondria, #13



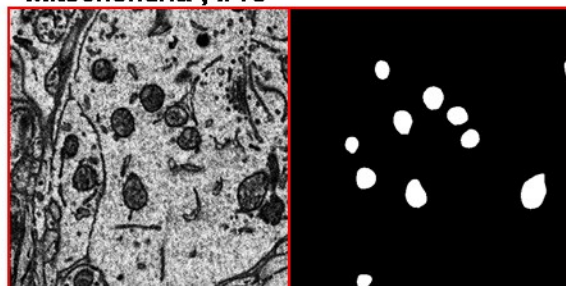
Mitochondria, #14



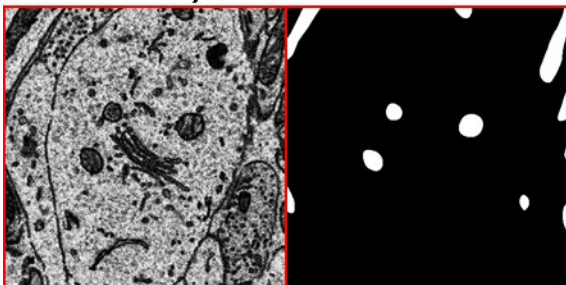
Mitochondria, #15



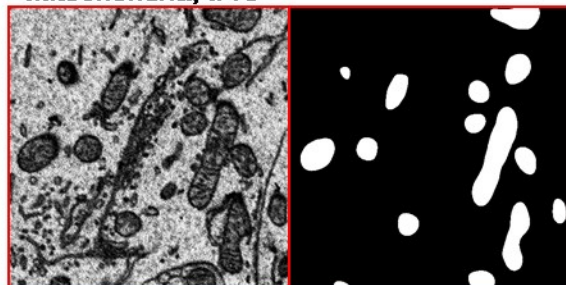
Mitochondria, #16



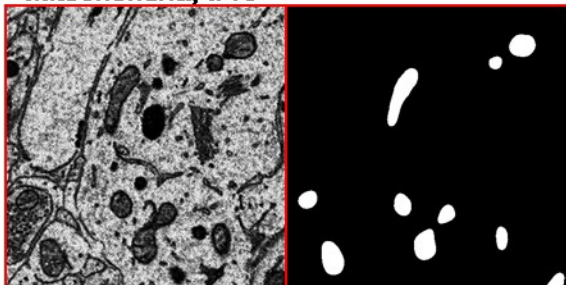
Mitochondria, #17



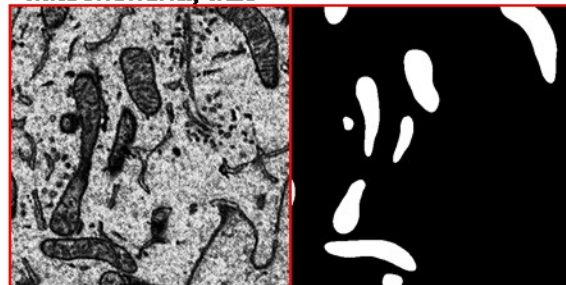
Mitochondria, #18



Mitochondria, #19

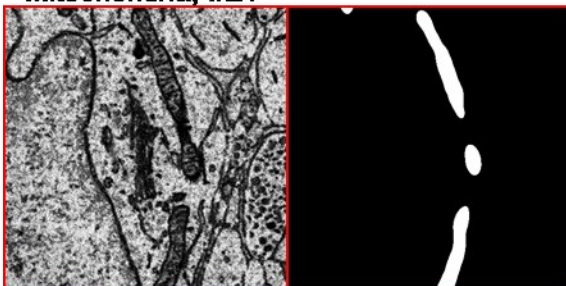


Mitochondria, #20

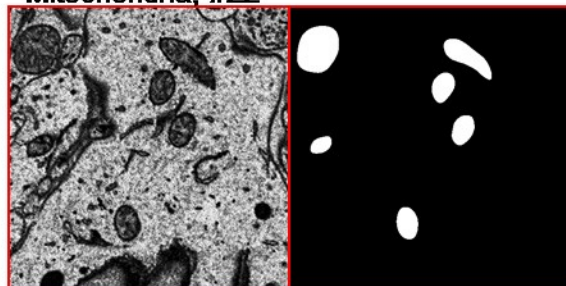




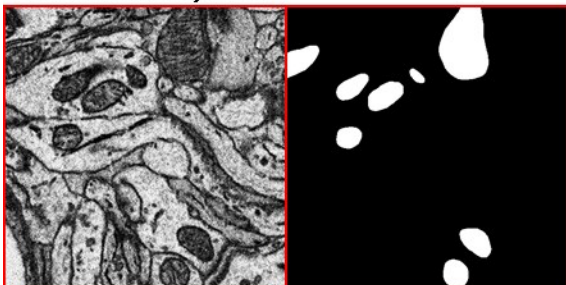
Mitochondria, #21



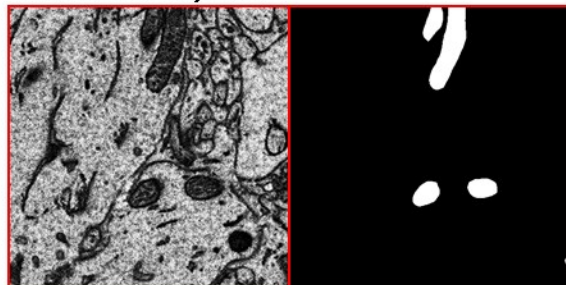
Mitochondria, #22



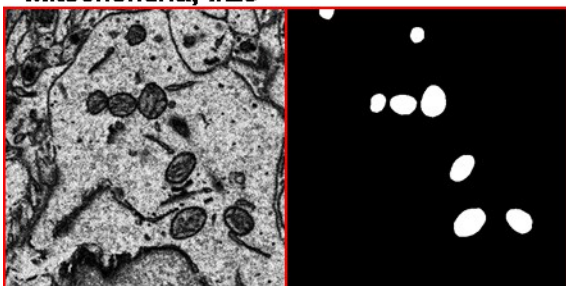
Mitochondria, #23



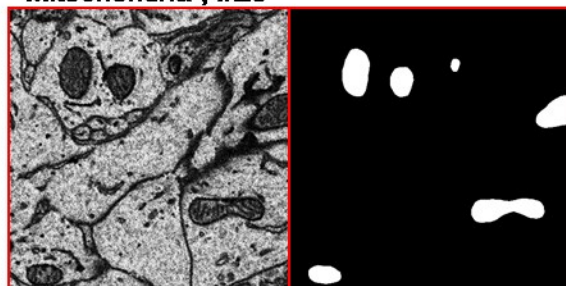
Mitochondria, #24



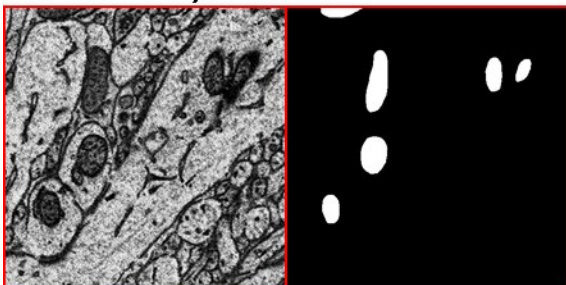
Mitochondria, #25



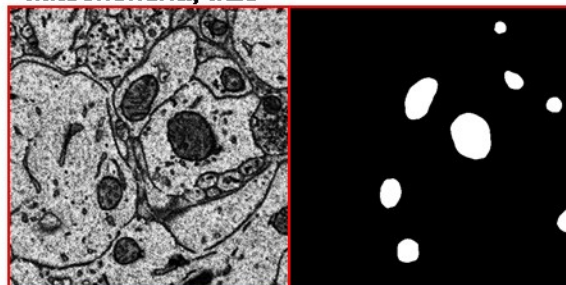
Mitochondria, #26



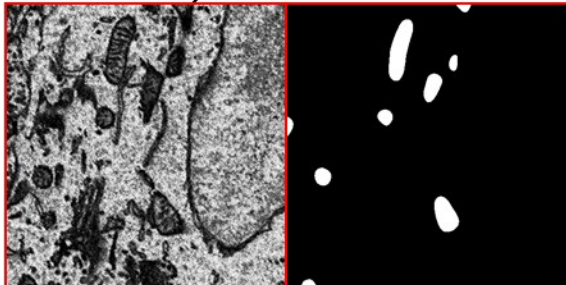
Mitochondria, #27



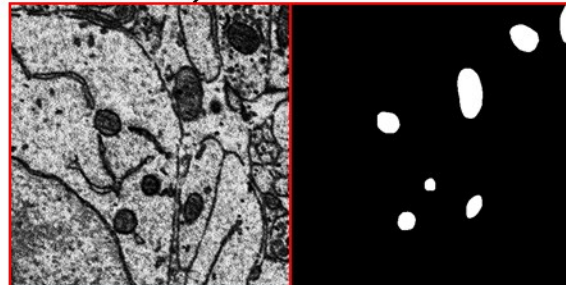
Mitochondria, #28



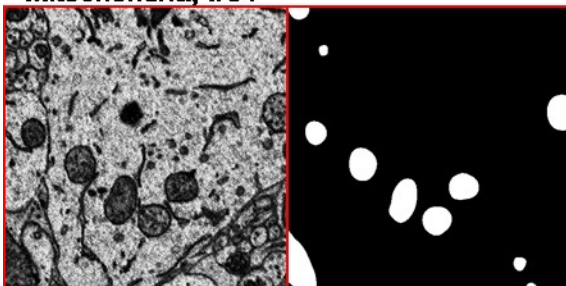
Mitochondria, #29



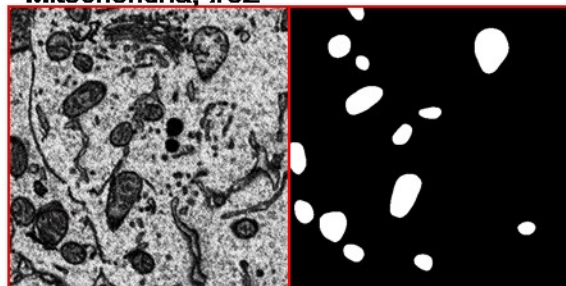
Mitochondria, #30



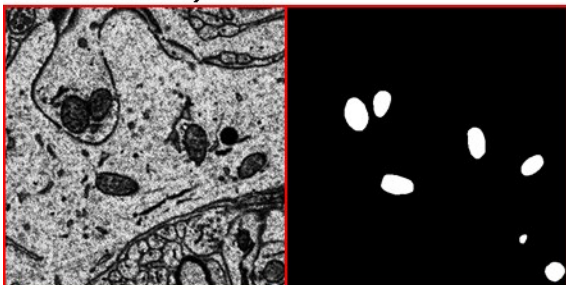
Mitochondria, #31



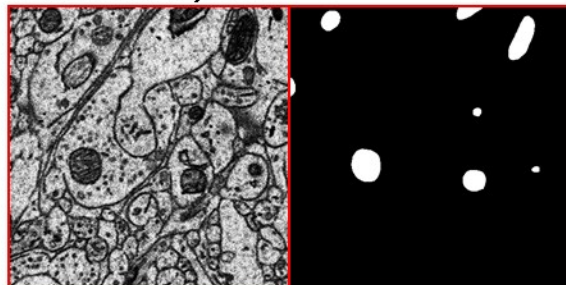
Mitochondria, #32



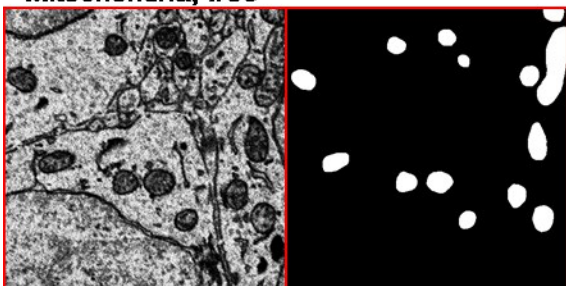
Mitochondria, #33



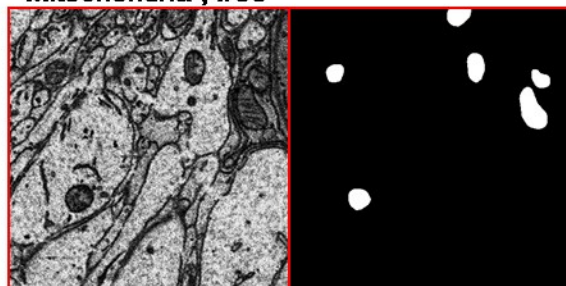
Mitochondria, #34



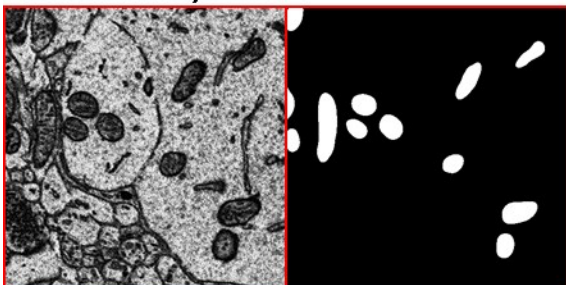
Mitochondria, #35



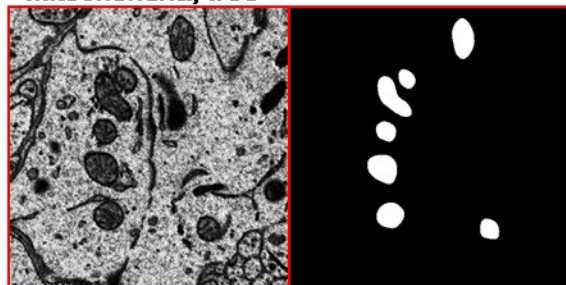
Mitochondria, #36



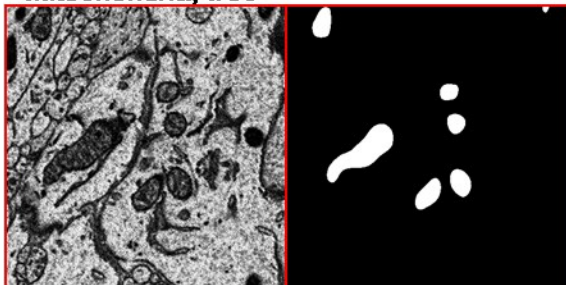
Mitochondria, #37



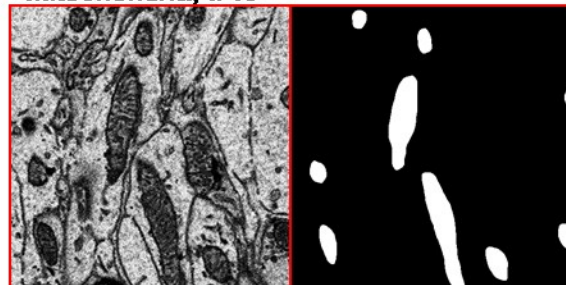
Mitochondria, #38



Mitochondria, #39

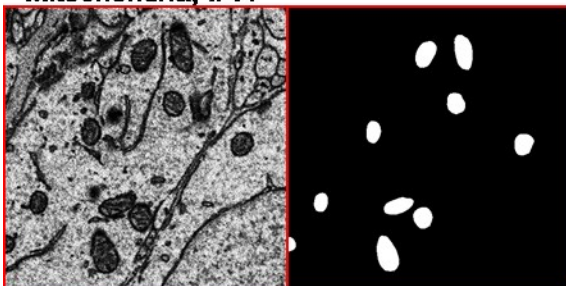


Mitochondria, #40

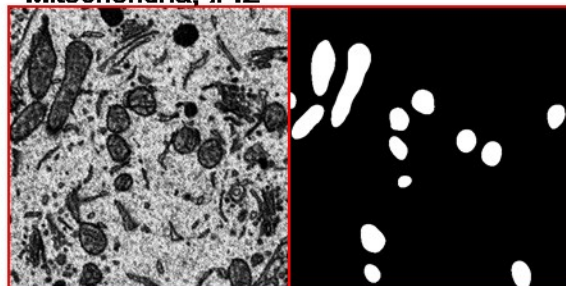




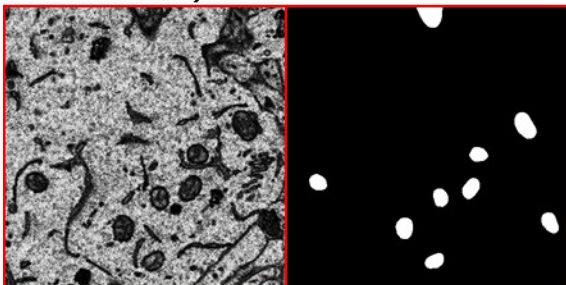
Mitochondria, #41



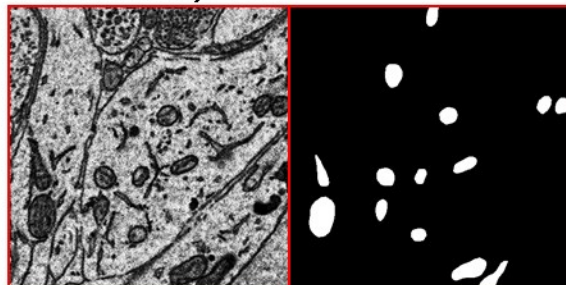
Mitochondria, #42



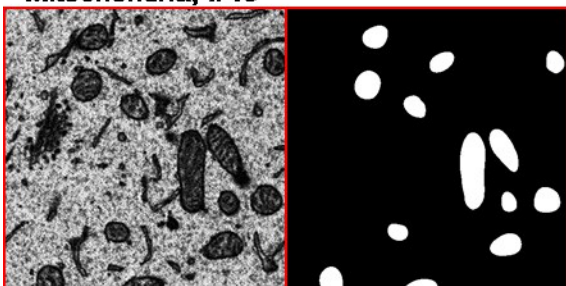
Mitochondria, #43



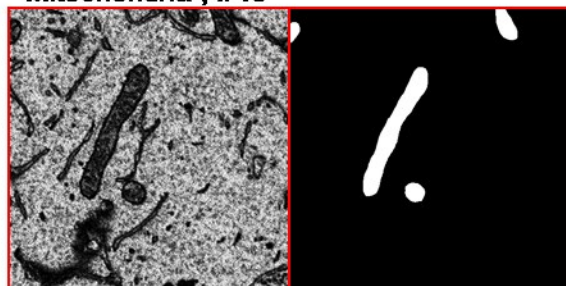
Mitochondria, #44



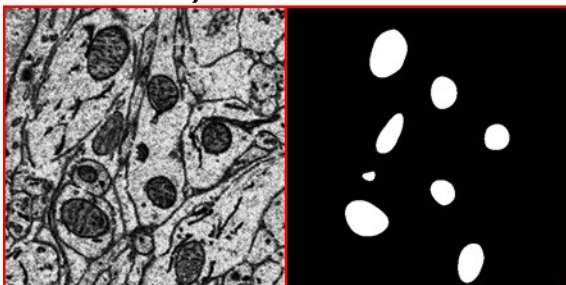
Mitochondria, #45



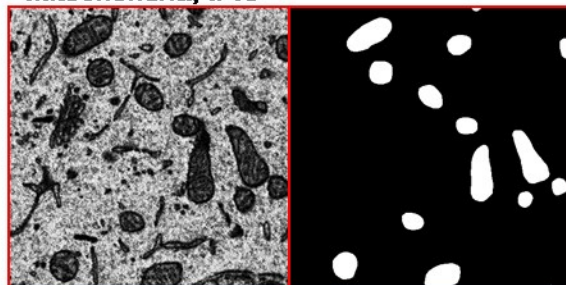
Mitochondria, #46



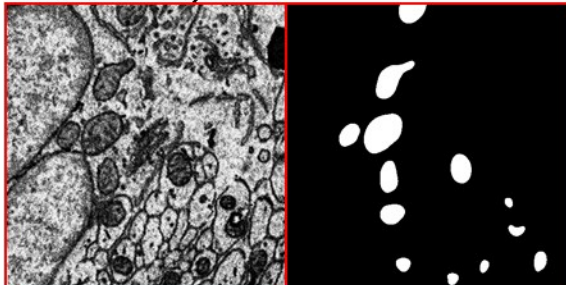
Mitochondria, #47



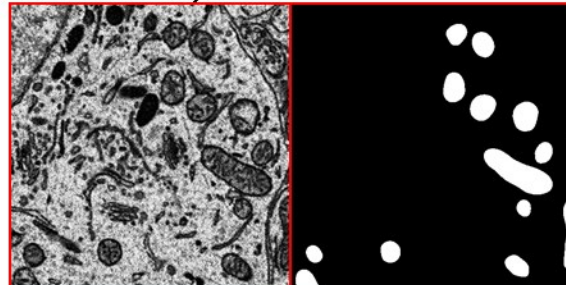
Mitochondria, #48



Mitochondria, #49



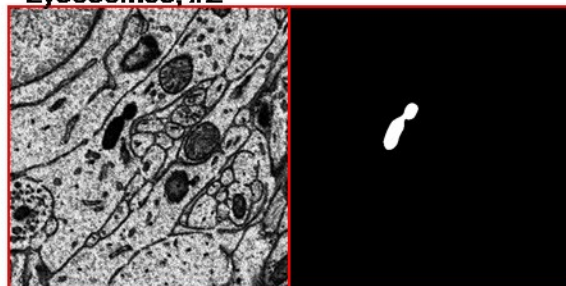
Mitochondria, #50



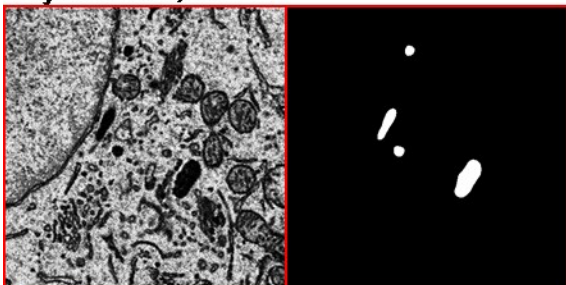
Lysosomes, #1



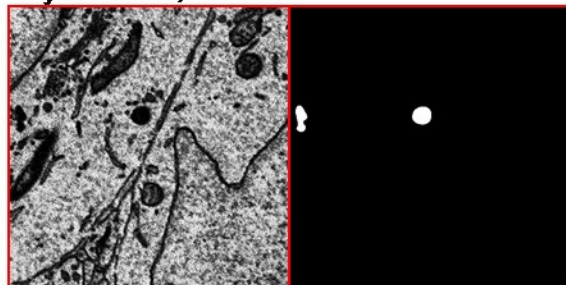
Lysosomes, #2



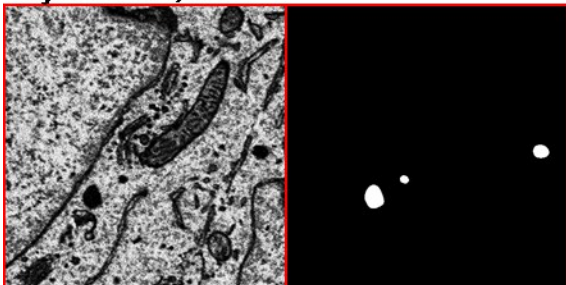
Lysosomes, #3



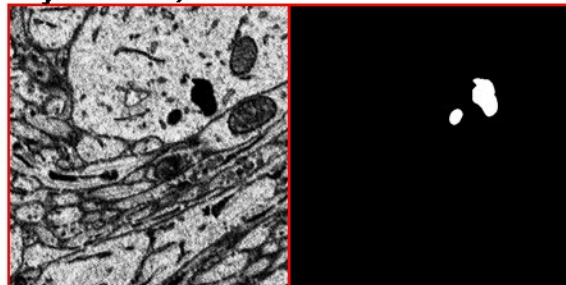
Lysosomes, #4



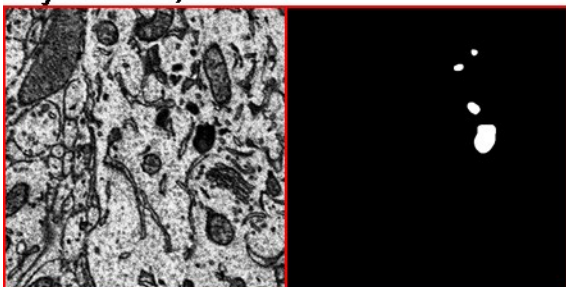
Lysosomes, #5



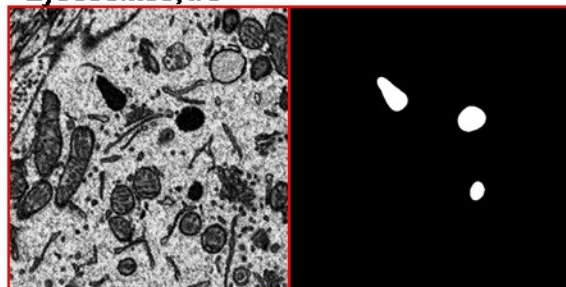
Lysosomes, #6



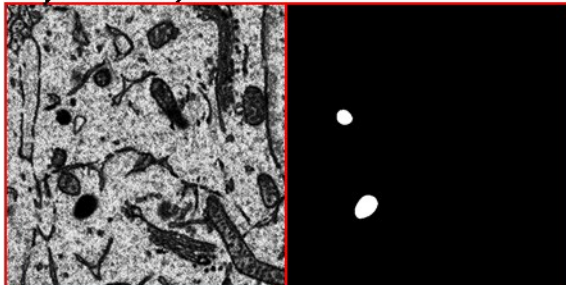
Lysosomes, #7



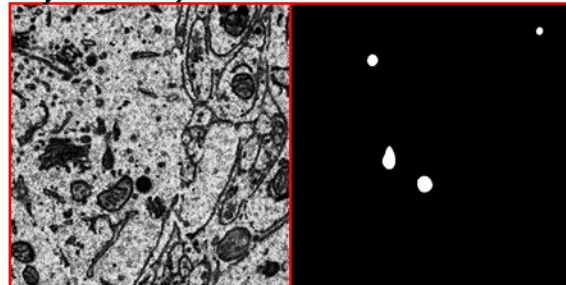
Lysosomes, #8



Lysosomes, #9

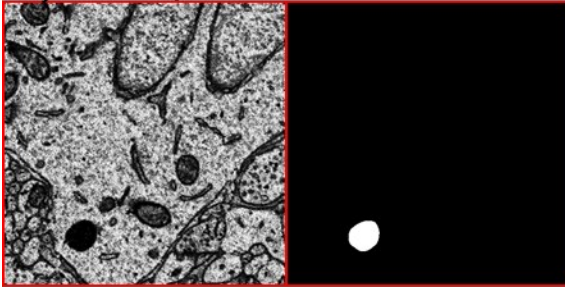


Lysosomes, #10

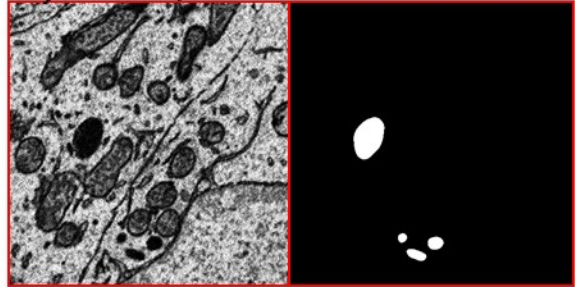




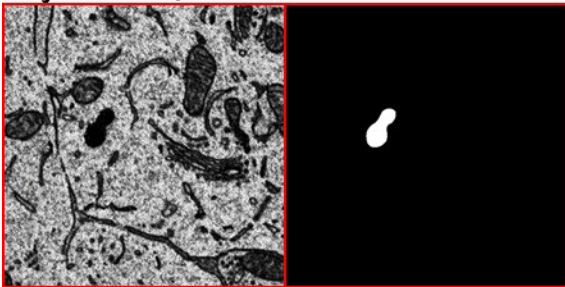
Lysosomes, #11



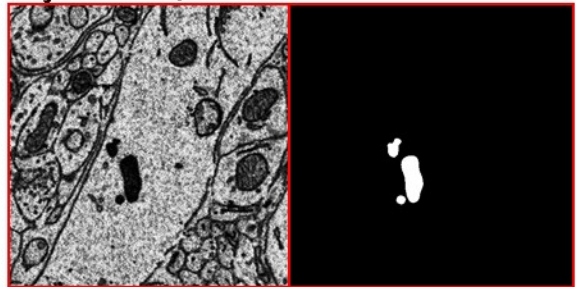
Lysosomes, #12



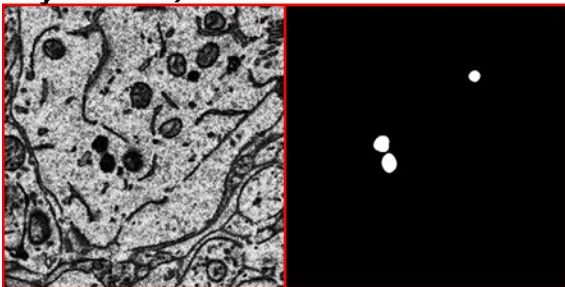
Lysosomes, #13



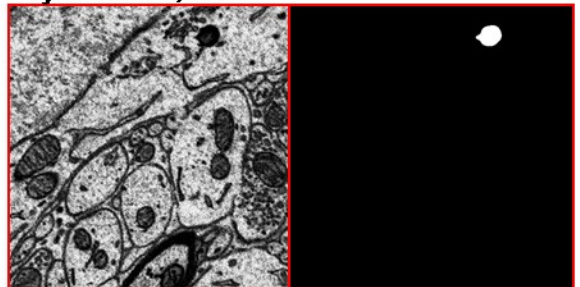
Lysosomes, #14



Lysosomes, #15



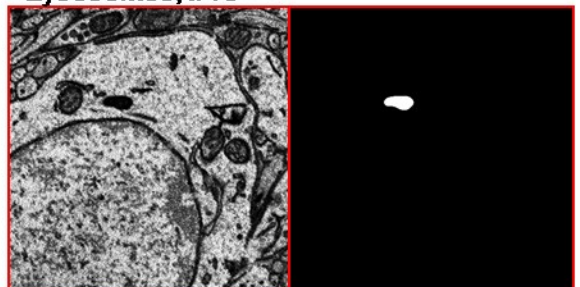
Lysosomes, #16



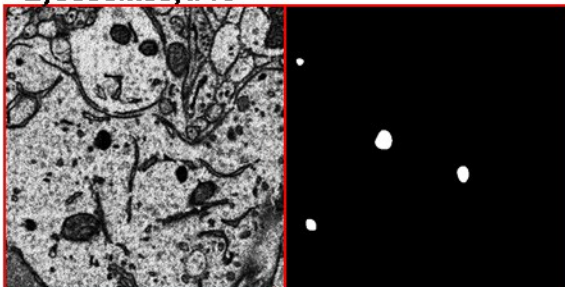
Lysosomes, #17



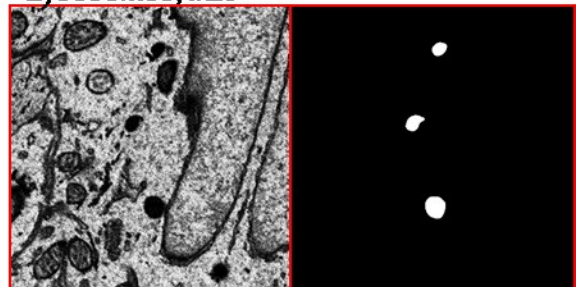
Lysosomes, #18



Lysosomes, #19



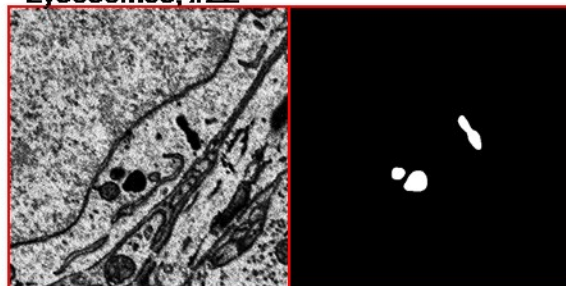
Lysosomes, #20



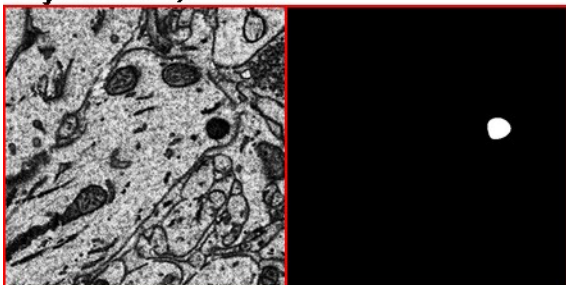
Lysosomes, #21



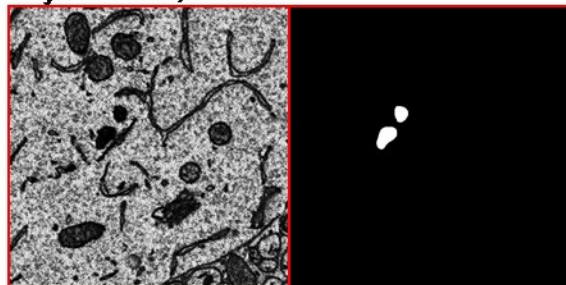
Lysosomes, #22



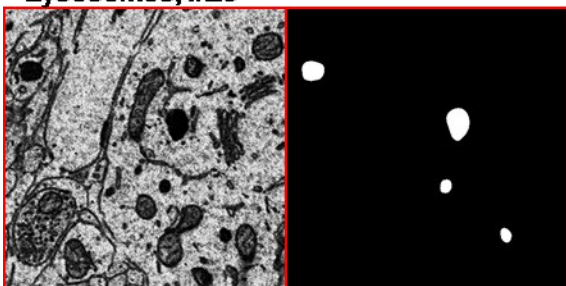
Lysosomes, #23



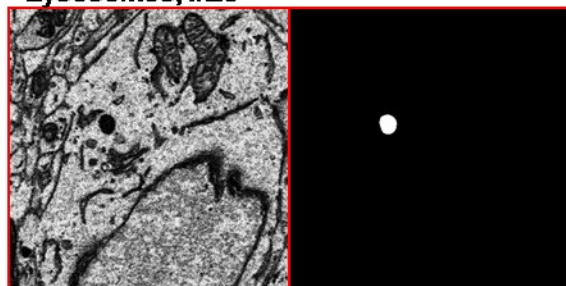
Lysosomes, #24



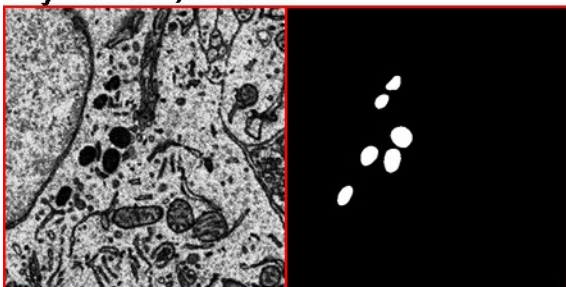
Lysosomes, #25



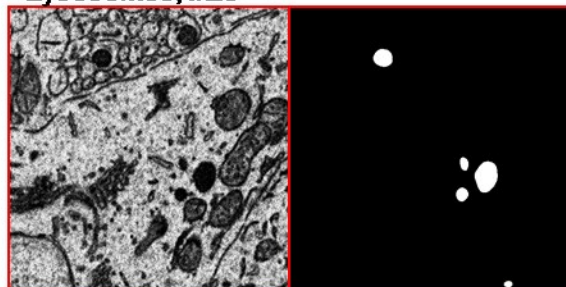
Lysosomes, #26



Lysosomes, #27



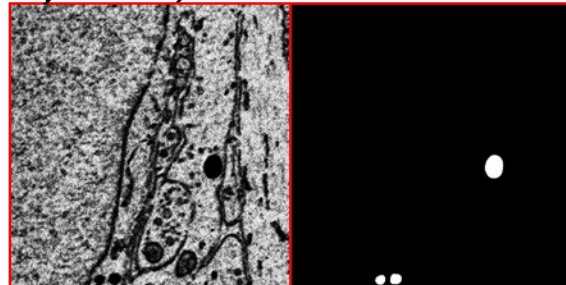
Lysosomes, #28



Lysosomes, #29

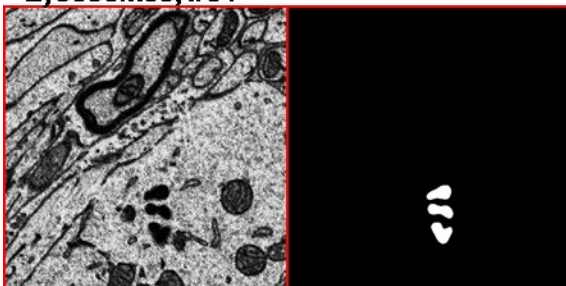


Lysosomes, #30

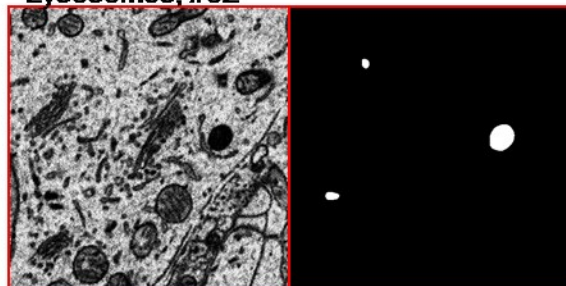




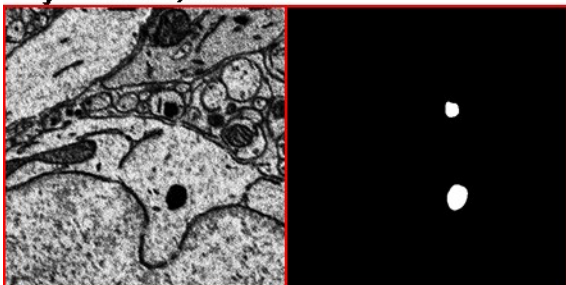
Lysosomes, #31



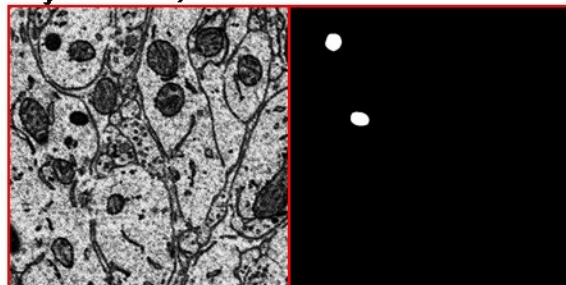
Lysosomes, #32



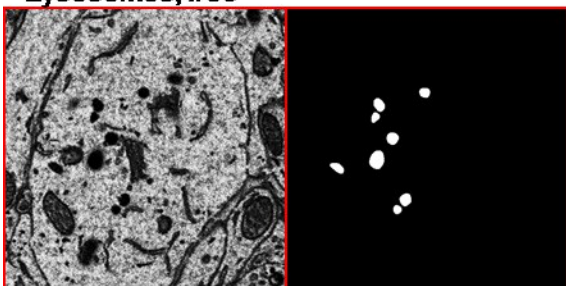
Lysosomes, #33



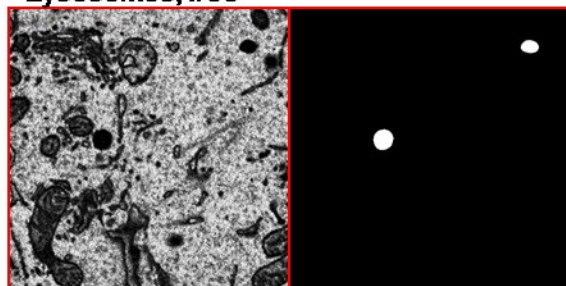
Lysosomes, #34



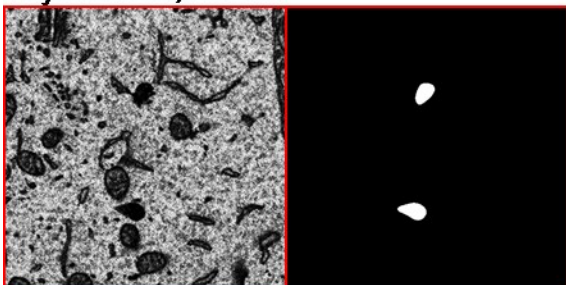
Lysosomes, #35



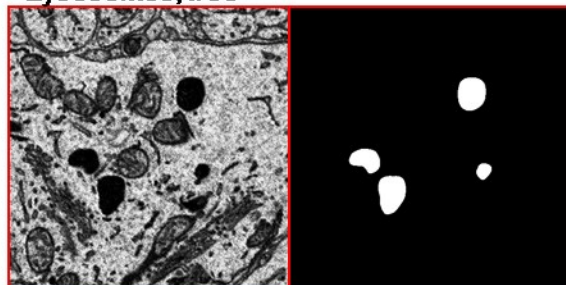
Lysosomes, #36



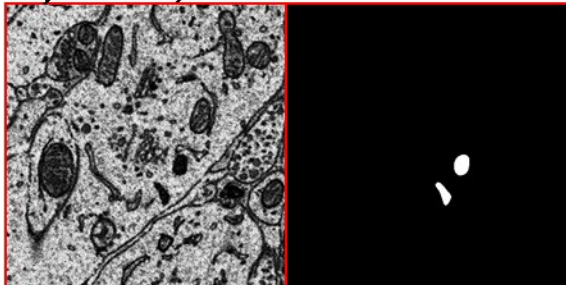
Lysosomes, #37



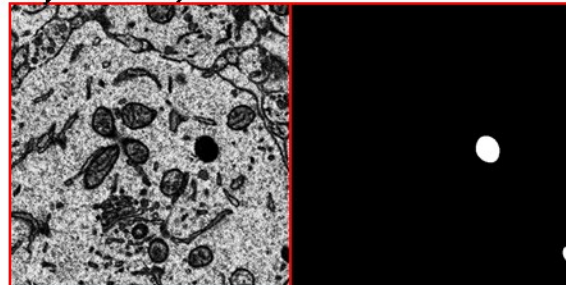
Lysosomes, #38



Lysosomes, #39



Lysosomes, #40



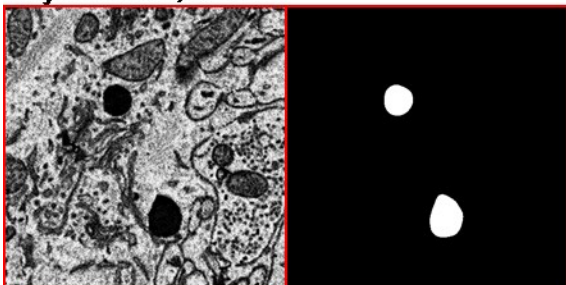
Lysosomes, #41



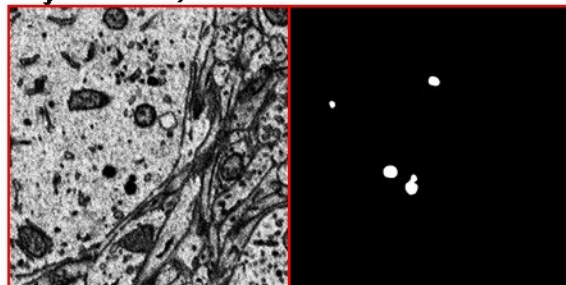
Lysosomes, #42



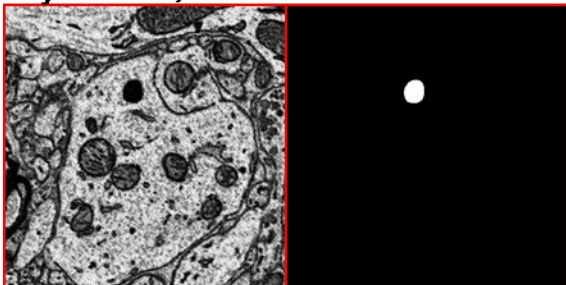
Lysosomes, #43



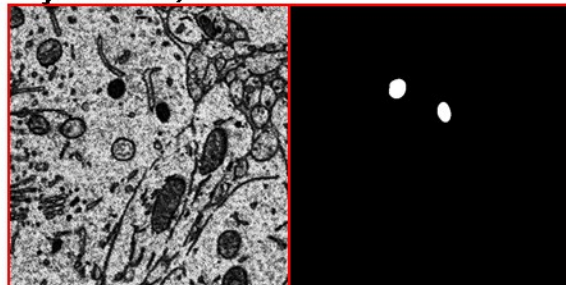
Lysosomes, #44



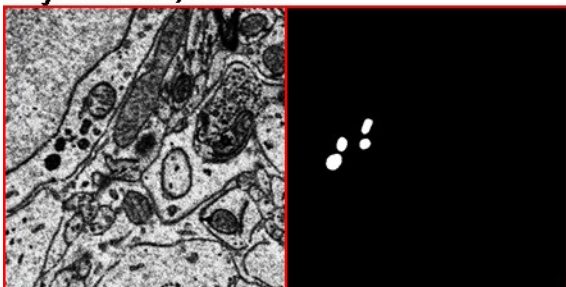
Lysosomes, #45



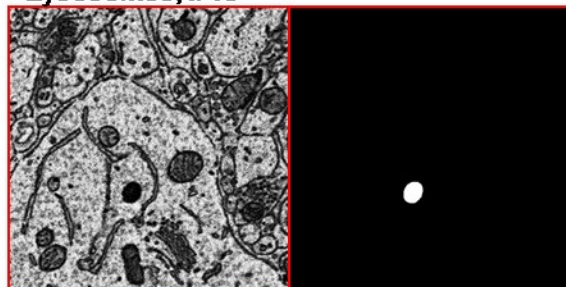
Lysosomes, #46



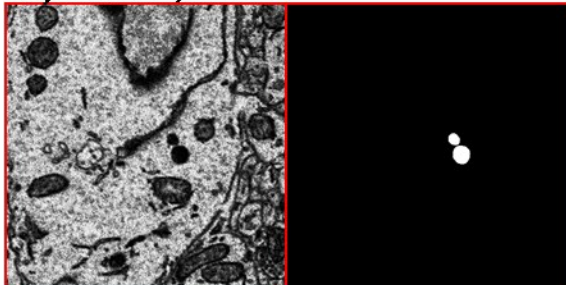
Lysosomes, #47



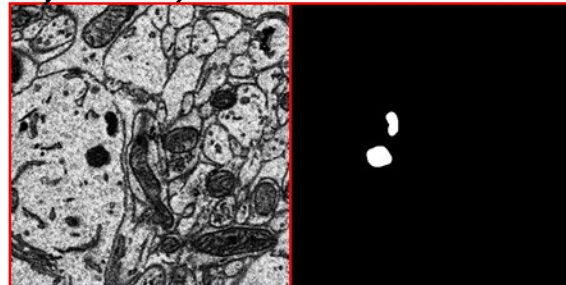
Lysosomes, #48



Lysosomes, #49



Lysosomes, #50



## Appendix C. Source Code

### C.2.1. newstack\_bin.sh

```
1      #!/bin/bash
2
3      function show_help () {
4      cat <<-END
5      newstack_bin.sh
6      Usage:
7      -----
8      -i | --input (MRC filename)
9          MRC stack to be binned
10
11      -o | --output (MRC filename)
12          Desired name of output MRC stack
13
14      -b | --bin (Integer)
15          Factor to bin the input stack by
16
17      -t | --time
18          If specified, timing stats for each step will be output
19
20      -h | --help
21          Display this help
22      END
23      }
24
25      while ;; do
26          case $1 in
27              -h|--help)
28                  show_help
29                  exit
30                  ;;
31              -i|--input)
32                  mrc_in=$2
33                  shift 2
34                  continue
35                  ;;
36              -o|--output)
37                  mrc_out=$2
38                  shift 2
39                  continue
40                  ;;
41              -b|--bin)
42                  bin=$2
43                  shift 2
44                  continue
45                  ;;
46              -t|--time)
47                  time=1
48                  shift 1
49                  continue
50                  ;;
```

```

51         *)
52         break
53     esac
54     shift
55 done
56
57 IMOD_BIN=${IMOD_DIR}/bin
58
59 #Check validity of input mrc stack
60 if [[ ! $mrc_in ]]; then printf 'ERROR: input stack not specified.\n\n' >&2; show_help; exit
61 1; fi
62 if [[ ! -f $mrc_in ]]; then printf 'ERROR: input stack does not exist.\n\n' >&2; show_help;
63 exit 1; fi
64 if [[ $IMOD_BIN/header $mrc_in > /dev/null
65 if [[ $? == 1 ]]; then printf 'ERROR: input stack is not a valid MRC file.\n\n' >&2;
66 show_help; exit 1; fi
67
68 #Check validity of output
69 if [[ ! $mrc_out ]]; then printf 'ERROR: output stack not specified.\n\n' >&2; show_help;
70 exit 1; fi
71 path_out=`dirname $mrc_out`
72 if [[ ! -d $path_out ]]; then printf 'ERROR: output directory does not exist.\n\n' >&2;
73 show_help; exit 1; fi
74
75 #Check validity of binning factor
76 if [[ ! $bin ]]; then printf 'ERROR: binning factor not specified.\n\n' >&2; show_help; exit 1;
77 fi
78 if [[ ${bin//[0-9]} ]]; then
79     CMD="${IMOD_BIN}/newstack -shrink ${bin}" #Use "-shrink" if the input is not an
80 integer
81 else
82     CMD="${IMOD_BIN}/newstack -bin ${bin}" #Use "-bin" if the input is an integer
83 fi
84
85 #Create temporary directory and log directory (if necessary)
86 mkdir ${path_out}/tmp_nb
87 if [[ -n $time ]]; then mkdir ${path_out}/log_nb; fi
88
89 #Get number of slices in the input stack
90 Nslices=`${IMOD_BIN}/header -size $mrc_in | tr -s ' ' | cut -d ' ' -f4`
91
92 #Loop over each slice. For each slice, first extract the given slice using "newstack -secs",
93 then bin the
94 #extracted slice using "newstack -bin". Record timing stats using /usr/bin/time if desired.
95 for ((i=0;i<=$(Nslices-1);i+=1)); do
96     N=`printf %04d $i`
97     file_extract=${path_out}/tmp_nb/slice_${N}.mrc
98     file_bin=${path_out}/tmp_nb/slice_bin_${N}.mrc
99     if [[ -n $time ]]; then
100         /usr/bin/time -v -o ${path_out}/log_nb/time_extract_${N}.txt ${IMOD_BIN}/newstack -
101 secs $i $mrc_in $file_extract
102         /usr/bin/time -v -o ${path_out}/log_nb/time_bin_${N}.txt $CMD $file_extract $file_bin
103     else
104         ${IMOD_BIN}/newstack -secs $i $mrc_in $file_extract

```

```

96     $CMD $file_extract $file_bin
97     fi
98     rm -rf $file_extract
99     done
100
101     #Append all single-slice, binned MRC files to the final stack
102     if [[ -n $time ]]; then
103         /usr/bin/time -v -o ${path_out}/log_nb/time_stack.txt ${IMOD_BIN}/newstack
104     else
105         ${IMOD_BIN}/newstack ${path_out}/tmp_nb/slice_bin*.mrc $mrc_out
106     fi
107
108     #Cleanup
109     rm -rf ${path_out}/tmp_nb

```

### C.2.2. mrcstack2png.sh

```

1     #! /bin/bash
2
3     function show_help () {
4         cat <<-END
5         mrcstack2png.sh
6         Usage:
7         -----
8         -i | --input (File name)
9             MRC stack to be converted to PNG files
10
11        -o | --output (Directory name)
12            Path to store output PNG files to
13
14        -a | --array
15            Process in parallel as an array job
16
17        -h | --help
18            Display this help
19    END
20    }
21
22    while ;; do
23        case $1 in
24            -h|--help)
25                show_help
26                exit
27                ;;
28            -i|--input)
29                input=$2
30                shift 2
31                continue
32                ;;
33            -o|--output)
34                path_out=$2
35                shift 2
36                continue

```



```

37         ;;
38     -a|--array)
39         array=1
40         shift 1
41         continue
42     ;;
43 *)
44     break
45 esac
46 shift
47 done
48
49 if [[ ! $input ]] || [[ ! $path_out ]]; then
50     printf 'ERROR: options -i and -o must be specified\n\n' >&2
51     show_help
52     exit 1
53 fi
54
55 source /home/aperez/.bashrc
56
57 if [[ ! -d $path_out ]]; then mkdir ${path_out}; fi
58 mkdir ${path_out}/test ${path_out}/log
59
60 #If array is not selected, launch a non-array job using standard mrc2tif. If it is set, launch
a parallel job
61 #using mrc2tif on single slices only.
62 if [[ -z ${array+x} ]]; then
63     qsub -v file_mrc=${input},path_out=${path_out} -o ${path_out}/log
/data/aperez/sge/mrcstack2png.q
64     else
65         Nslices=`${IMOD_DIR}/bin/header -size $input | tr -s ' ' | cut -d ' ' -f4`
66         qsub -t 1-${Nslices} -v file_mrc=${input},path_out=${path_out} -o ${path_out}/log
/data/aperez/sge/mrcstack2png.q
67     fi

```

### C.2.3. mrcstack2png.q

```

1     #! /bin/bash
2
3     #$ -S /bin/bash
4     #$ -N mrc2png
5     #$ -j yes
6     #$ -m eas
7     #$ -M alexjperez@outlook.com
8     #$ -l h_vmem=1G
9     #$ -cwd
10    #$ -V
11
12    source /home/aperez/.bashrc
13
14    base=${file_mrc%.mrc}
15    if [[ $SGE_TASK_ID -eq 0 ]]; then
16        ${IMOD_DIR}/bin/mrc2tif -p ${file_mrc} ${path_out}/test/${base}
17    else

```

```

18     ${IMOD_DIR}/bin/mrc2tif -p -z $((SGE_TASK_ID-1)),$((SGE_TASK_ID-1)) ${file_mrc}
${path_out}/test/${base}
19     if [[ $((SGE_TASK_ID-1)) -lt 1000 ]]; then
20         ext=`printf %03d $((SGE_TASK_ID-1))`
21     else
22         ext=$((SGE_TASK_ID-1))
23     fi
24     mv ${path_out}/test/${base}.${ext}.png ${path_out}/test/${base}.`printf %04d
$((SGE_TASK_ID-1))`.png
25     fi

```

#### C.2.4. find\_nonborder\_pixels.m

```

1     function [PV,border] = find_nonborder_pixels( I )
2     % Returns the pixel values of an input image that are not part
3     % of the border. The border is determined by computing the
4     % gradient magnitude of the image, then searching for pixels
5     % with gradient values that are approximately zero. This method
6     % of determining the image border works for all types of borders,
7     % including simple linear translations and shears.
8     %
9     % INPUT
10    % -----
11    % I      Image to extract pixel values from
12    %
13    % OUTPUT
14    % -----
15    % PV      1xM vector of non-border pixel values
16    % border  Binary image displaying the border pixels
17    %
18
19    [FX,FY] = gradient(double(I));
20    border = sqrt(FX.^2 + FY.^2); %Gradient magnitude
21    border = (border < 0.01); %Find where gradient is ~zero
22    border = imfill(~border,'holes');
23    clear FX FY
24
25    RP = regionprops(border,I,'PixelValues');
26    PV = []; %Initialize pixel value vector
27    for i = 1:numel(RP)
28        PV = [PV RP(i).PixelValues'];
29    end
30
31    end

```

#### C.2.5. generate\_reference.sh

```

1     #! /bin/bash
2
3     function show_help () {
4     cat <<-END
5     generate_reference.sh
6     Usage:
7     -----

```

```

8     -i | --input (Directory name)
9         Path containing the stack of PNG files to run EHS on
10
11     -o | --output (Directory name)
12         Path to store the reference histogram to
13
14     -f | --fullstack
15         Compute the reference histogram as that of the full image stack specified by --input
16
17     -z (Integer)
18         Compute the reference histogram as that of a single image in --input whose value is
19         specified here
20
21     -h | --help
22         Display this help
23     END
24 }
25 while ;; do
26     case $1 in
27         -h|--help)
28             show_help
29             exit
30             ;;
31         -i|--input)
32             path_in=$2
33             shift 2
34             continue
35             ;;
36         -o|--output)
37             path_out=$2
38             shift 2
39             continue
40             ;;
41         -f|--fullstack)
42             fullstack=1
43             shift 1
44             continue
45             ;;
46         -z)
47             z=$2
48             shift 2
49             continue
50             ;;
51         *)
52             break
53     esac
54     shift
55 done
56
57 if [[ ! $path_in ]] || [[ ! $path_out ]]; then
58     printf 'ERROR: options -i and -o must be specified\n\n' >&2
59     show_help; exit 1
60 fi

```



```

61
62 if [[ ! $fullstack ]] && [[ ! $z ]]; then
63     printf 'ERROR: must chose fullstack mode (-f) or single image mode (-z integer)\n\n'
>&2
64     show_help; exit 1
65 fi
66
67 if [[ ! -d $path_in ]]; then
68     printf 'ERROR: directory specified by -i does not exist\n\n' >&2
69     show_help; exit 1
70 fi
71
72 if [[ ! -d $path_out ]]; then mkdir ${path_out}; fi
73 mkdir ${path_out}/log ${path_out}/err ${path_out}/ref
74
75 if [[ -z ${fullstack+x} ]]; then
76     qsub -v path_in=${path_in},path_out=${path_out}/ref,N1=${z} -o ${path_out}/log
generate_reference.q
77 else
78     Nslices=`ls ${path_in} | wc -l`
79     qsub -t 1-${Nslices} -v path_in=${path_in},path_out=${path_out}/ref,N1=1 -o
${path_out}/log -e ${path_out}/err generate_reference.q
80 fi

```

### C.2.6. generate\_reference.q

```

1     #!/bin/bash
2
3     #$ -S /bin/bash
4     #$ -N genRef
5     #$ -m eas
6     #$ -M alexjperez@outlook.com
7     #$ -l h_vmem=5G
8     #$ -cwd
9     #$ -V
10
11    if [[ $SGE_TASK_ID -ne 0 ]]; then N1=${SGE_TASK_ID}; fi
12
13    matlab -nodisplay -nosplash -r "generate_reference('${path_in}','${path_out}',${N1})";

```

### C.2.7. generate\_reference.m

```

1     function generate_reference( path_in, path_out, N )
2     % Generate a reference histogram for exact histogram specification
3     % Generates a histogram with 256 bins for an input 8-bit image. Prior
4     % to histogram calculation, borders are removed from the input image.
5     % The output histogram is stored to an ASCII text file.
6     %
7     % INPUT
8     % -----
9     % path_in   Path to the stack of TIF/PNG files to process
10    % path_out  Path to output the histogram text file to
11    % N        Image number to analyze within path_in
12    %

```

```

13
14 %Parse path_in for test images
15 imgs = dir(fullfile(path_in, '*.png'));
16 if isempty(imgs)
17     imgs = dir(fullfile(path_in, '*.tif'));
18 end
19
20 file_out = fullfile(path_out, ['ref_hist_' sprintf('%04d', N) '.txt']);
21
22 %Compute overall histogram. Each image is read, and its borders are removed.
23 %The image histogram is computed only from the pixels remaining following
24 %border removal.
25 file_in = fullfile(path_in, imgs(N).name);
26 img_in = uint8(imread(file_in));
27 [IR IC] = size(img_in);
28 [PV_img, ~] = find_nonborder_pixels(img_in);
29 clear img_in
30 fprintf('Analyzing %s\n', file_in);
31 fprintf('Image size = %d\n', IR*IC);
32 fprintf('Border size = %d\n', IR*IC - numel(PV_img));
33 hist_img = hist(double(PV_img), 256);
34 hist_img = hist_img';
35
36 save(file_out, 'hist_img', '-ascii');
37 fprintf('Output written to %s\n', file_out);
38
39 end

```

### C.2.8. run\_ehs.sh

```

1    #!/bin/bash
2
3    function show_help () {
4    cat <<-END
5    run_ehs.sh
6    Usage:
7    -----
8    -i | --images (Directory name)
9        Path containing input PNG files to be processed
10
11    -r | --reference (Directory name)
12        Path containing reference histogram text files
13
14    -o | --output (Directory name)
15        Path to save output to
16
17    -h | --help
18        Display this help
19    END
20    }
21
22    while ;; do
23        case $1 in
24            -h|--help)

```

```

25         show_help
26         exit
27         ;;
28     -i|--images)
29         path_images=$2
30         shift 2
31         continue
32         ;;
33     -r|--reference)
34         path_ref=$2
35         shift 2
36         continue
37         ;;
38     -o|--output)
39         path_out=$2
40         shift 2
41         continue
42         ;;
43     *)
44         break
45     esac
46     shift
47 done
48
49 if [[ ! $path_images ]] || [[ ! $path_ref ]] || [[ ! $path_out ]]; then
50     printf 'ERROR: options -i, -r, and -o must be specified\n\n' >&2
51     show_help
52     exit 1
53 fi
54
55 if [[ ! -d $path_out ]]; then mkdir ${path_out}; fi
56 mkdir ${path_out}/log ${path_out}/ehs
57
58 Nslices=`ls ${path_images} | wc -l`
59 qsub -t 1-${Nslices} -v
path_images=${path_images},path_ref=${path_ref},path_out=${path_out}/ehs -o ${path_out}/log
run_ehs.q

```

### C.2.9. run\_ehs.q

```

1     #! /bin/bash
2
3     #$ -S /bin/bash
4     #$ -N runEHS
5     #$ -j yes
6     #$ -m eas
7     #$ -M alexjperez@outlook.com
8     #$ -l h_vmem=15G
9     #$ -cwd
10    #$ -V
11
12    matlab -nodisplay -nosplash -r
"run_ehs("${path_images}","${path_ref}","${path_out}","${SGE_TASK_ID}");

```

**C.2.10. run\_ehs.m**

```

1     function run_ehs( path_imgs, path_ref, path_out, N )
2
3     files_ref = dir(fullfile(path_ref, '*.txt'));
4     imgs = dir(fullfile(path_imgs, '*.png'));
5     if isempty(imgs)
6         imgs = dir(fullfile(path_imgs, '*.tif'));
7     end
8
9     %Compute summed histogram of all references
10    hist_sum = zeros(256,1);
11    for i = 1:numel(files_ref)
12        hist_i = load(fullfile(path_ref,files_ref(i).name));
13        hist_sum = hist_sum + hist_i;
14    end
15
16    I = uint8(imread(fullfile(path_imgs,imgs(N).name)));
17    [PV_init,border] = find_nonborder_pixels(I);
18
19    [ehs,~] = exact_histogram(I,hist_sum,border);
20    clear I border
21
22    [PV_ehs,~] = find_nonborder_pixels(ehs);
23
24    for i = 1:256
25        fprintf('%d %d %d %d\n',i-1,hist_sum(i),PV_init(i),PV_ehs(i));
26    end
27
28    %Write output
29    [~,base,ext] = fileparts(imgs(N).name);
30    file_out = fullfile(path_out,[base '_EHS' ext]);
31    imwrite(uint8(ehs),file_out);
32    fprintf('Output written to %s\n',file_out);
33
34    end

```

**C.2.11. process\_td.sh**

```

1     #! /bin/bash
2
3     function show_help () {
4     cat <<-END
5     process_td.sh
6     Usage:
7     -----
8     -i | --input (File name)
9         MRC file the training contours were traced on
10
11    -m | --model (File name)
12        Model file consisting of two objects. The first object consists of scattered seed points
13        marking the center of each training image. The second object consists of closed
contours
14        representing manual traces of the object of interest.

```

```
15
16     -e | --ehs (Directory name)
17         Path to output stack of PNGs from EHS
18
19     -o | --output (Directory name)
20         Path to save training images and labels to
21
22     -d | --dim (Integer,Integer)
23         Dimensions of the training data in X and Y
24
25     -h | --help
26         Display this help
27 END
28 }
29
30 while ;; do
31     case $1 in
32         -h|--help)
33             show_help
34             exit
35             ;;
36         -i|--input)
37             file_mrc=$2
38             shift 2
39             continue
40             ;;
41         -m|--model)
42             file_mod=$2
43             shift 2
44             continue
45             ;;
46         -e|--ehs)
47             path_ehs=$2
48             shift 2
49             continue
50             ;;
51         -o|--output)
52             path_out=$2
53             shift 2
54             continue
55             ;;
56         -d|--dim)
57             dim=$2
58             shift 2
59             continue
60             ;;
61         *)
62             break
63     esac
64     shift
65 done
66
67 source /home/aperez/.bashrc
68
```

```

69  if [[ ! $file_mrc ]] || [[ ! $file_mod ]] || [[ ! $path_ehs ]] || [[ ! $path_out ]] || [[ ! $dim ]]; then
70      printf 'ERROR: options -i, -m, -e, -o, and -d must be specified\n\n' >&2
71      show_help
72      exit 1
73  fi
74
75  if [[ ! -d $path_ehs ]]; then
76      printf 'ERROR: the path specified by -e does not exist\n\n' >&2
77      show_help
78      exit 1
79  fi
80
81  if [[ ! -f $file_mrc ]]; then
82      printf 'ERROR: the MRC file specified by -i does not exist\n\n' >&2
83      show_help
84      exit 1
85  fi
86
87  if [[ ! -f $file_mod ]]; then
88      printf 'ERROR: the model file specified by -m does not exist\n\n' >&2
89      show_help
90      exit 1
91  fi
92
93  Nobj=`${IMOD_DIR}/bin/imodinfo -a $file_mod | grep -m 1 '^imod' | cut -d ' ' -f2`
94
95  if [[ $Nobj -ne 2 ]]; then
96      printf 'ERROR: the model file specified by -m must contain exactly two objects\n\n' >&2
97      show_help
98      exit 1
99  fi
100
101  if [[ ! -d $path_out ]]; then mkdir ${path_out}; fi
102  mkdir ${path_out}/td ${path_out}/tl ${path_out}/log
103
104  qsub -v
file_mrc=${file_mrc},file_mod=${file_mod},path_ehs=${path_ehs},path_out=${path_out},dim=${di
m} -o ${path_out}/log process_td.q

```

### C.2.12. process\_td.q

```

1  #! /bin/bash
2
3  # $ -S /bin/bash
4  # $ -N processTD
5  # $ -j yes
6  # $ -m eas
7  # $ -M alexjperez@outlook.com
8  # $ -l h_vmem=2G
9  # $ -cwd
10 # $ -V
11
12 source /home/aperez/.bashrc
13

```

```

14  img_h=`${IMOD_DIR}/bin/header -size ${file_mrc} | tr -s ' ' | cut -d ' ' -f3`
15
16  dimx=`echo $dim | cut -d ' ' -f1`
17  dimy=`echo $dim | cut -d ' ' -f2`
18  radx=`echo "$dimx / 2" | bc`
19  rady=`echo "$dimy / 2" | bc`
20
21  #Extract seed and contour model files
22  file_seed=${file_mod%.mod}_seed.mod
23  file_cont=${file_mod%.mod}_cont.mod
24
25  ${IMOD_DIR}/bin/imodextract 1 $file_mod $file_seed
26  ${IMOD_DIR}/bin/imodextract 2 $file_mod $file_cont
27
28  #Output point listing of seed file to a text file
29  ${IMOD_DIR}/bin/model2point $file_seed ${file_seed%.mod}.txt
30
31  C=1
32  while read line; do
33      #Extract point values from text file
34      td_i=td_`printf '%03d' $C`
35      tl_i=tl_`printf '%03d' $C`
36      xi=`echo $line | cut -d ' ' -f1`
37      yi=`echo $line | cut -d ' ' -f2`
38      zi=`echo $line | cut -d ' ' -f3`
39
40      #Determine bounding box
41      xmin=`echo "$xi - $radx" | bc`
42      xmax=`echo "$xi + $radx - 1" | bc`
43      ymin=`echo "$yi - $rady" | bc`
44      ymax=`echo "$yi + $rady - 1" | bc`
45
46      #Trim the bounding box from the input mrc stack. Generate training labels with
47      imodmop ${IMOD_DIR}/bin/trimvol -x ${xmin},${xmax} -y ${ymin},${ymax} -z $((zi+1)),$((zi+1))
48      $file_mrc ${tl_i}.mrc
49      ${IMOD_DIR}/bin/imodmop -mask 1 $file_cont ${tl_i}.mrc ${tl_i}.mrc
50      ${IMOD_DIR}/bin/mrc2tif -p ${tl_i}.mrc ${path_out}/tl/${tl_i}.png
51      rm -rf ${tl_i}*
52
53      #Generate training data from EHS output
54      file_td=`ls ${path_ehs} | sed -n "$((zi+1))p"
55      /home/aperez/usr/local/bin/convert ${path_ehs}/${file_td} -crop
56      ${dimx}x${dimx}+${xmin}+${((img_h-ymax))} ${path_out}/td/${td_i}.png
57
58      C=$((C+1))
59
60  done < ${file_seed%.mod}.txt
61
62  #Clean up intermediates
63  rm -rf $file_seed $file_cont ${file_seed%.mod}.txt

```

**C.2.13. CHM\_array\_testTile.q**

```

1      #!/bin/bash
2
3      # $ -V
4      # $ -cwd
5      # $ -j y
6      # $ -S /bin/bash
7      # $ -m eas
8      # $ -M alexjperez@outlook.com
9      # $ -N CHM
10     # $ -l h_vmem=10G
11     # $ -t 1-1283:1
12
13     #####
14     Dir_IM=/home/aperez/usr/local/bin
15     classdir=/data/aperez/CHM_classifiers/nucleolus/ZT04/edge/Nstage2_Nlevel2
16     testfolder=/data/aperez/ZT04/input_iso_full/XY/histeq/PNG
17     testout=/data/aperez/ZT04/runCHM_nucleolus/XY
18     Nstage=2
19     Nlevel=2
20     TileX=8
21     TileY=6
22     Overlap=200
23     #####
24
25     img_in=`ls ${testfolder}/*.png | sed -n "${SGE_TASK_ID}";${SGE_TASK_ID}p`
26
27     base=`basename ${img_in}`
28     base=${base%.*}
29
30     testTile=${testfolder}/${base}
31     mkdir ${testTile}
32
33     ${Dir_IM}/convert ${img_in} -crop ${TileX}x${TileY}+${Overlap}+${Overlap}@! +repage
+adjoin ${testTile}/${base}_tile_%04d.png
34
35     testF=""${testTile}""
36     testO=""${testout}""
37     matlab -nodisplay -singleCompThread -r
'TrainScript_test('${testF},${testO},${Nstage},${Nlevel}'); quit'
38
39     rm -rf ${testTile} #Remove input tiles
40
41     #####
42     # Stitching
43     #####
44
45     pwd
46     cd ../output_testImages
47
48     DX=${TileX} #Number of tiles in X
49     DY=${TileY} #Number of tiles in Y
50     OX=${Overlap} #Overlap in X in pixels

```



```

51     OY=${Overlap} #Overlap in Y in pixels
52
53     if (( ${OX} == 0 & ${OY} == 0 )); then #Perform simple stitching if no overlap was
specified
54         ${Dir_IM}/montage ${base}_tile*.png -mode concatenate -tile ${DX}x${DY}
${base}.png
55         ${Dir_IM}/convert ${base}.png -equalize ${base}.png
56         exit 0
57     fi
58
59     # First, the row is stitched together in a west-to-east (left-to-right) manner. Rows are then
stitched together in a
60     # north-to-south (top-to-bottom) manner. Regions of overlap are handled by taking the
maximum pixel value in the
61     # region
62     C=1
63     for ((j=1;j<=${DY};j+=1)); do #Loop over rows
64         imgFirst=`ls ${base}_tile*.png | sed -n "${C}","${C}'p` #Name of first image in the row
65         imgHeight=`${Dir_IM}/identify -format "%[fx:h]" ${imgFirst}` #Height of row
66         for ((i=1;i<=${DX};i+=1)); do #Loop over each image within the row
67             imgL=`ls ${base}_tile*.png | sed -n "${C}","${C}'p` #Name of first image
68             imgLWidth=`${Dir_IM}/identify -format "%[fx:w]" ${imgL}`
69             if (( i == 1 )); then #If first image in the row, no overlap is needed in the western
direction
70                 imgR=`ls ${base}_tile*.png | sed -n "${(C+1)}","${(C+1)}'p` #Name of second
image
71                 imgRWidth=`${Dir_IM}/identify -format "%[fx:w]" ${imgR}`
72                 ${Dir_IM}/convert ${imgL} -gravity west -crop -${OX}-0 ${base}_tempL.png
#Crop non-overlap region of image1
73                 ${Dir_IM}/convert ${imgL} -gravity east -crop ${OX}x${imgHeight}-0-0
${base}_tempML.png #Crop eastern overlap region of image1
74                 ${Dir_IM}/convert ${imgR} -gravity west -crop ${OX}x${imgHeight}-0-0
${base}_tempMR.png #Crop western overlap region of image2
75                 ${Dir_IM}/convert ${base}_tempML.png ${base}_tempMR.png -compose lighten -
composite ${base}_tempM.png #Take max of overlap regions
76                 ${Dir_IM}/convert +append ${base}_tempL.png ${base}_tempM.png
${base}_out_temp.png #Append
77                 elif (( i > 1 & i <= ${DX-1} )); then #All middle images in the row
78                 imgR=`ls ${base}_tile*.png | sed -n "${(C+1)}","${(C+1)}'p`
79                 imgRWidth=`${Dir_IM}/identify -format "%[fx:w]" ${imgR}`
80                 ${Dir_IM}/convert ${imgL} -gravity center -crop ${imgLWidth-
2*OX}x${imgHeight}-0-0 ${base}_tempL.png #Crop non-overlap region of image1
81                 ${Dir_IM}/convert ${imgL} -gravity east -crop ${OX}x${imgHeight}-0-0
${base}_tempML.png #Crop eastern overlap region of image1
82                 ${Dir_IM}/convert ${imgR} -gravity west -crop ${OX}x${imgHeight}-0-0
${base}_tempMR.png #Crop western overlap region of image2
83                 ${Dir_IM}/convert ${base}_tempML.png ${base}_tempMR.png -compose
lighten -composite ${base}_tempM.png #Take max of overlap regions
84                 ${Dir_IM}/convert +append ${base}_out_temp.png ${base}_tempL.png
${base}_tempM.png ${base}_out_temp.png #Append
85                 else #Last image in the row, overlap has already been analyzed
86                 ${Dir_IM}/convert ${imgL} -gravity east -crop -${OX}-0 ${base}_tempL.png #Crop
non-overlap region

```

```

87             ${Dir_IM}/convert +append ${base}_out_temp.png ${base}_templ.png
${base}_out_temp.png #Append
88         fi
89         rm -rf ${base}_temp*.png
90         C=$((C+1))
91     done
92     if (( j == 1 )); then #If first row, rename to out.png
93         imgWidthTotal=`${Dir_IM}/identify -format "%[fx:w]" ${base}_out_temp.png`
94         mv ${base}_out_temp.png ${base}_out.png
95     else
96         imgHeightOut=`${Dir_IM}/identify -format "%[fx:h]" ${base}_out.png` #Get height of
total output to this point
97         ${Dir_IM}/convert ${base}_out.png -gravity south -crop ${imgWidthTotal}x${OY}-
0-0 ${base}_tempMU.png
98         ${Dir_IM}/convert ${base}_out.png -gravity north -crop
${imgWidthTotal}x$((imgHeightOut-OY))-0-0 ${base}_tempU.png
99         ${Dir_IM}/convert ${base}_out_temp.png -gravity north -crop
${imgWidthTotal}x${OY}-0-0 ${base}_tempMD.png
100        ${Dir_IM}/convert ${base}_out_temp.png -gravity south -crop
${imgWidthTotal}x$((imgHeightOut-OY))-0-0 ${base}_tempD.png
101        ${Dir_IM}/convert ${base}_tempMU.png ${base}_tempMD.png -compose lighten
-composite ${base}_tempM.png
102        ${Dir_IM}/convert -append ${base}_tempU.png ${base}_tempM.png
${base}_tempD.png ${base}_out.png
103    fi
104    rm -rf ${base}_temp*.png ${base}_out_temp*.png
105 done
106
107    rm -rf ${base}_tile*.png #Remove output tiles

```

#### C.2.14. segStats.m

```

1    function fid_out = segStats( Seg, GT, Border, Filename )
2
3    % segStats
4    % Computes a variety of segmentation evaluation statistics comparing a
5    % single probability map or a 3D stack of probability maps to a single
6    % binary ground truth image or a 3D stack of binary ground truth images.
7    % Works with both:
8    %     (1) Pre-imported image stacks
9    %     (2) A directory specifying the location of images to load
10   % If the input probability maps are already binary because they have
11   % been pre-processed, statistics will be computed directly between this
12   % stack and the ground truth. If not, statistics will be computed at
13   % variable threshold levels, ranging from 0-255 in increments of 1. The
14   % metrics calculated are reported, in order, as follows:
15   %
16   %     (1) Threshold Value
17   %     (2) Error Probability [11]
18   %     (3) False Positive Rate (a.k.a. fall-out) [1]
19   %     (4) False Negative Rate (a.k.a. miss rate) [1]
20   %     (5) True Positive Rate (a.k.a. sensitivity, recall) [1]
21   %     (6) True Negative Rate (a.k.a. specificity) [1]
22   %     (7) Negative Predictive Value [1]

```

```

23 % (8) False Discovery Rate [1]
24 % (9) Precision [1]
25 % (10) Rand Accuracy [1]
26 % (11) F-value (a.k.a. F-score, F-measure) [1]
27 % (12) Jaccard Similarity Coefficient [2,3]
28 % (13) Dice Coefficient [9,10]
29 % (14) Geometric Mean (a.k.a. G-Mean) [8]
30 % (15) Matthew's Correlation Coefficient [4,5]
31 % (16) Average Conditional Probability [5,6]
32 % (17) Area Under Curve [2]
33 % (18) Balanced Accuracy [7]
34 % (19) Informedness [2]
35 % (20) Markedness [2]
36 % (21) False Positives (total pixels)
37 % (22) False Negatives (total pixels)
38 % (23) True Positives (total pixels)
39 % (24) True Negatives (total pixels)
40 % (25) FP + FN + TP + TN (as a sanity check, should be equal to Image Size)
41 % (26) Image Size (total pixels)
42 %
43 % Input
44 % -----
45 % Seg      Single image or 3D stack of probability map images. There
46 %          are two use cases:
47 %          (1) Image stack has been previously loaded. In this case,
48 %          Seg is the name of the matrix containing the stack.
49 %          (2) Image stack needs to be loaded. In this case, Seg is
50 %          a string specifying the directory of the files to load.
51 % GT       Single image or 3D stack of binary ground truth. The use
52 %          cases are the same as Seg.
53 % Border   Integer value specifying the padding to remove around both
54 %          the Seg and GT stacks.
55 % Filename Path to write a text file containing the output statistics
56 %          to.
57 %
58 % Example
59 % -----
60 % segStats(Out,GT,0,'stats/segStats_Out.txt');
61 % segStats('/home/aperez/out','home/aperez/gt',50,'/home/aperez/stats.txt');
62 %
63 % References
64 % -----
65 % [1] Fawcett, T. (2006). An introduction to ROC analysis. Pattern recognition
66 %     letters, 27(8), 861-874.
67 % [2] Powers, D. M. (2011). Evaluation: from precision, recall and F-measure to
68 %     ROC, informedness, markedness & correlation. Journal of Machine Learning
69 %     Technologies, 2(1), 37-63.
70 % [3] Lucchi, A., Smith, K., Achanta, R., Knott, G., & Fua, P. (2012). Supervoxel-
71 %     based segmentation of mitochondria in EM image stacks with learned shape
72 %     features. Medical Imaging, IEEE Transactions on, 31(2), 474-486
73 % [4] Matthews, B. W. (1975). Comparison of the predicted and observed secondary
74 %     structure of T4 phage lysozyme. Biochimica et Biophysica Acta (BBA)-Protein
75 %     Structure, 405(2), 442-451.
76 % [5] Baldi, P., Brunak, S., Chauvin, Y., Andersen, C. A., & Nielsen, H. (2000).

```

```

77 % Assessing the accuracy of prediction algorithms for classification: an
78 % overview. Bioinformatics, 16(5), 412-424.i
79 % [6] Burset, M., & Guigo, R. (1996). Evaluation of gene structure prediction
80 % programs. Genomics, 34(3), 353-367.
81 % [7] Brodersen, K. H., Ong, C. S., Stephan, K. E., & Buhmann, J. M. (2010).
82 % The balanced accuracy and its posterior distribution. In Pattern Recognition
83 % (ICPR), 2010 20th International Conference on (pp. 3121-3124). IEEE.
84 % [8] Seyedhosseini, M., Sajjadi, M., & Tasdizen, T. (2013). Image Segmentation with
85 % Cascaded Hierarchical Models and Logistic Disjunctive Normal Networks.
Computer
86 % Vision.
87 % [9] Shattuck, D. W., Prasad, G., Mirza, M., Narr, K. L., & Toga, A. W. (2009).
88 % Online resource for validation of brain segmentation methods. NeuroImage,
89 % 45(2), 431-439.
90 % [10] Dice, L. R. (1945). Measures of the amount of ecologic association between
91 % species. Ecology, 26(3), 297-302.
92 % [11] Celebi, M. E., Schaefer, G., Iyatomi, H., Stoecker, W. V., Malters, J. M., &
93 % Grichnik, J. M. (2009). An improved objective evaluation measure for border
94 % detection in dermoscopy images. Skin Research and Technology, 15(4), 444-450.
95 %
96
97 if nargin < 4; Filename = ""; end
98 if nargin < 3; Border = 0; end
99
100 % Import probability map. If the input is not a string, assume it is a matrix to which
101 % the probability map images have been previously imported to. If it is a string,
102 % check if it is a directory of a file. If it is a directory, import all images in the
103 % directory to a 3D image stack. If not, import the single image.
104 if isstr(Seg) & isdir(Seg)
105     Dir_seg = Seg;
106     Imgs_seg = dir([Dir_seg '/*.png']);
107     if isempty(Imgs_seg)
108         Imgs_seg = dir([Dir_seg '/*.tif']);
109     end
110     Img_seg = imread([Dir_seg '/' Imgs_seg(1).name]);
111     Seg = zeros([size(Img_seg) numel(Imgs_seg)]);
112     clear Img_seg
113     for i = 1:numel(Imgs_seg)
114         Seg(:, :, i) = imread([Dir_seg '/' Imgs_seg(i).name]);
115     end
116 elseif isstr(Seg) & ~isdir(Seg)
117     Seg = imread(Seg);
118 end
119 Seg = uint8(Seg);
120
121 % Import ground truth stack in the same fashion.
122 if isstr(GT) & isdir(GT)
123     Dir_gt = GT;
124     Imgs_gt = dir([Dir_gt '/*.png']);
125     if isempty(Imgs_gt)
126         Imgs_gt = dir([Dir_gt '/*.tif']);
127     end
128     Img_gt = imread([Dir_gt '/' Imgs_gt(1).name]);
129     GT = zeros([size(Img_gt) numel(Imgs_gt)]);

```

```

130     clear lmg_gt
131     for i = 1:numel(lmgs_gt)
132         GT(:, :, i) = imread([Dir_gt '/' lmgs_gt(i).name]);
133     end
134 elseif isstr(GT) & ~isdir(GT)
135     GT = imread(GT);
136 end
137
138 % Remove borders.
139 [SegX SegY SegZ] = size(Seg);
140 [GTX GTY GTZ] = size(GT);
141 Seg = Seg(Border+1:SegX-Border, Border+1:SegY-Border, :);
142 GT = GT(Border+1:GTX-Border, Border+1:GTY-Border, :);
143 [SegX SegY SegZ] = size(Seg);
144 N = SegX * SegY * SegZ;
145
146 % Open file to write output to
147 if isempty(Filename)
148     fid_out = 1;
149 else
150     fid_out = fopen(Filename, 'a');
151 end
152
153 % Check if Seg is binary (i.e. the probability map has already been
154 % thresholded). If so, set Tmax to 0. If not, set Tmax to 255, so
155 % statistics will be calculated at each value of T.
156 if numel(unique(Seg)) <= 2
157     Tmax = 0;
158 else
159     Tmax = 255;
160 end
161
162 % Calculate statistics
163 T = [0:Tmax];
164 Seg_orig = Seg;
165 for i = 1:numel(T)
166     Seg = (Seg_orig > T(i));
167     fprintf('Calculating statistics for T = %d\n', T(i));
168     FP = numel(find( Seg == 1 & GT == 0 )); % [1], False positive OR Type I Error
169     FN = numel(find( Seg == 0 & GT == 1 )); % [1], False negative OR Type II Error
170     TP = numel(find( Seg == 1 & GT == 1 )); % [1], True positive OR Hit
171     TN = numel(find( Seg == 0 & GT == 0 )); % [1], True negative OR Correct Rejection
172     FPR(i,1) = FP/(FP+TN); % [1], False positive rate OR Fall-out
173     FNR = FN/(FN+TP); % [1], False negative rate OR Miss rate
174     TPR(i,1) = TP/(TP+FN); % [1], True positive rate OR Sensitivity OR Recall OR Hit Rate
175     TNR = TN/(FP+TN); % [1], True negative rate OR Specificity
176     NPV = TN/(TN+FN); % [1], Negative Predictive Value (NPV)
177     FDR = FP/(FP+TP); % [1], False Discovery Rate (FDR)
178     Precision = TP/(TP+FP); % [1], Precision OR Positive Predictive Value (PPV)
179     F1 = (2*TP)/(2*TP+FP+FN); % [1], F1-score OR F-value OR F-measure
180     Accuracy = (TP+TN)/(TP+FN+FP+TN); % [1], Rand Accuracy
181     Jaccard = TP/(FP+TP+FN); % [2,3], Jaccard similarity coefficient OR VOC score
182     MCC = (TP*TN-FP*FN)/sqrt((TP+FP)*(TP+FN)*(TN+FP)*(TN+FN)); % [4,5], Matthew's
correlation coefficient

```

```

183     ACP = 0.25*((TP/(TP+FN))+(TP/(TP+FP))+(TN/(TN+FP))+(TN/(TN+FN))); % [5,6],
Average conditional probability
184     AUC = 1-((FPR(i,1)+FNR)/2); % [2], Area under curve
185     Inform = TPR(i,1)+TNR-1; % [2], Informedness
186     Marked = Precision+NPV-1; % [2], Markedness
187     AccBal = 0.5*(TPR(i,1)+TNR); % [7], Balanced accuracy
188     GMean = sqrt(TPR(i,1)*TNR); % [8], Geometrical mean OR G-Mean
189     Dice = (2*TP)/((FP+TP)+(TP+FN)); % [9,10], Dice Coefficient
190     ErrorProb = (FP+FN)/(TP+FN+FP+TN); % [11], Error Probability
191
192     fprintf(fid_out,'%d %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %d
%d %d %d %d %d\n',...
193     T(i),ErrorProb,FPR(i,1),FNR,TPR(i,1),TNR,NPV,FDR,Precision,Accuracy,F1,Jaccard,Dice,GMean,MCC,ACP,...
194     AUC,AccBal,Inform,Marked,FP,FN,TP,TN,(FP+FN+TP+TN),N);
195     end
196
197     % Calculate the area under the curve for the ROC plot
198     %[FPR_sort,I] = sort(FPR,1,'ascend');
199     %TPR_sort = TPR(I);
200     %AUC = trapz(FPR_sort,TPR_sort);
201     %fprintf(fid_out,'%f\n',AUC);
202
203
204     fclose(fid_out);
205
206     end

```

### C.2.15. binarize\_pm\_activecontour.m

```

1     function [seg,mask,Q] = binarize_pm_activecontour( path_in, path_out, N, level, iter,
smooth )
2     % Segments an input probability map by using the Chan-Vese method for active contour
evolution
3     % with seeds determined by multi-level Otsu thresholding. Requires MATLAB R2013a or
later for
4     % the functions multithresh, imquantize, and activecontour.
5     %
6     % INPUT
7     % -----
8     % path_in   Path containing probability maps to process
9     % path_out  Path to write segmented images to
10    % N         Image number within path_in to process
11    % level     Number of Otsu threshold levels to use for seed masking
12    % iter      Number of iterations of active contour evolution to use
13    % smooth    Degree of smoothing for active contour evolution
14    %
15    % OUTPUT
16    % -----
17    % seg       Output segmentation of the probability map specified by path_in and N
18    % mask      Binary mask used as seeds for active contour evolution
19    % Q         Image of the probability map after multi-level Otsu thresholding. Pixel values
20    %           will be integers from [1,...,level+1].

```

```

21 %
22 % USAGE
23 % -----
24 % Helpful values of iter typically range from 50 - 400, depending on the feature
segmented.
25 % Helpful values of level typically range from 2-10, depending on the feature
segmented and
26 % the quality of the probability map. Noisy probability maps with higher noise will
typically
27 % benefit more from increased levels. Helpful values of smooth range from 2-12.
28 %
29 % EXAMPLE
30 % -----
31 % [seg,mask,Q] = binarize_pm_activecontour('probMaps','seg',100,2,100,4);
32 %
33 % REFERENCES
34 % -----
35 % [1] Perez, A.J., Seyedhosseini, M., Deerinck, T.J., Bushong, E.A., Panda, S.,
Tasdizen, T.,
36 % and Ellisman, M.H. (2014). A workflow for the automatic segmentation of
organelles in
37 % electron microscopy image stacks. Frontiers in Neuroanatomy (in review).
38 % [2] Chan, T.F., and Vese, L.A. (2001). Active contours without edges. IEEE
Transactions on
39 % Image Processing, 10(2), 266-277.
40 %
41
42 tic;
43
44 if ~isdir(path_out); mkdir(path_out); end
45
46 imgs = dir(fullfile(path_in, '*.tif'));
47 if isempty(imgs)
48     imgs = dir(fullfile(path_in, '*.png'));
49 end
50
51 % Single-slice probability maps output from CHM will sometimes have a third dimension,
where each
52 % slice along the third dimension is identical. If this is the case, reduce the dimensionality
53 % of the input.
54 img = imread(fullfile(path_in, imgs(1).name));
55 if ndims(img) == 3; img = img(:,:,1); end
56 seg = zeros(size(img)); %Initialize empty output image
57 clear img
58
59 file_in = fullfile(path_in, imgs(N).name); %Read probability map N within path_in
60 P = double(imread(file_in));
61 if ndims(P) == 3; P = P(:,:,1); end %Reduce dimensionality if necessary
62 P = P / 255; %Rescale from [0,...,255] to [0,..., 1]
63 fprintf('Input image %s read.\n', file_in);
64
65 fprintf('Generating seed image via Otsu multi-level thresholding with ');
66 fprintf('%d levels.\n', level);
67 thresh = multithresh(P, level); %Return multi-level threshold values

```

```

68 Q = imquantize(P,thresh); %Apply threshold values to the probability map
69 mask = ( Q == level+1 ); %Mask Q at the top level of the threshold
70 mask = bwmorph(mask,'shrink',2); %Shrink the mask slightly to serve as seeds
71
72 fprintf('Performing Chan-Vese active contour segmentation with %d ',iter);
73 fprintf('iterations and a smoothing factor of %d.\n',smooth);
74 seg = activecontour(P,mask,iter,'Chan-Vese',smooth); %Active contour evolution
75
76 file_out = fullfile(path_out,['out_' sprintf('%04d',N) '.tif']);
77 imwrite(uint8(seg),file_out); %Write output in uint8 format
78 fprintf('Output image %s written.\n',file_out);
79 fprintf('Elapsed time: %0.2f\n',toc);
80
81 end

```

### C.2.XX. myelin\_segment\_sbem.q

```

1   #!/bin/bash
2
3   # $ -V
4   # $ -cwd
5   # $ -j y
6   # $ -S /bin/bash
7   # $ -m eas
8   # $ -M alexjperez@outlook.com
9   # $ -N myelinSeg
10  # $ -l h_vmem=2G
11  # $ -t 1-1000:1
12
13  matlab -nodisplay -nosplash -r
"myelin_segment_sbem('TIF_2Dbin10','out_2Dbin10',{SGE_TASK_ID},2,0)";

```

### C.3.1. mpas.sh

```

1   #!/bin/bash
2
3   function show_help () {
4   cat <<-END
5   mpas.sh
6   Usage:
7   -----
8   -i | --input (Dir1,Dir2,...,DirN)
9       Paths to probability maps in orientations 1,2,...,N
10
11  -o | --output (Directory name)
12      Path to store output and temporary files to
13
14  -r | --orientation (String1,String2,...,StringN)
15      Orientations of the probability maps in paths 1,2,...,N
16
17  -s | --size (Integer,Integer,Integer)
18      Size of the input stack (X,Y,Z) in the XY orientation
19
20  -h | --help

```



```

21     Display this help
22 END
23 }
24
25 while ;; do
26     case $1 in
27         -h|--help)
28             show_help
29             exit
30             ;;
31         -i|--input)
32             paths=$2
33             shift 2
34             continue
35             ;;
36         -o|--output)
37             path_out=$2
38             shift 2
39             continue
40             ;;
41         -s|--size)
42             size=$2
43             shift 2
44             continue
45             ;;
46         -r|--orientation)
47             orientations=$2
48             shift 2
49             continue
50             ;;
51         *)
52             break
53     esac
54     shift
55 done
56
57 #Make output directory (if necessary) and temporary directories
58 if [[ ! -d $path_out ]]; then mkdir $path_out; fi
59 mkdir ${path_out}/tmp ${path_out}/log
60
61 #Parse the inputs to determine how many axes have been specified by the user
62 size=`echo $size | tr ' ' ','`
63 Npath=`echo $paths | tr -cd , | wc -c`
64 Npath=$((Npath+1))
65 Nori=`echo $orientations | tr -cd , | wc -c`
66 Nori=$((Nori+1))
67 if [[ $Npath -ne $Nori ]]; then printf 'ERROR: # of paths and orientations must be the
same\n\n' >&2; show help; exit 1; fi
68
69 #Loop over all orientations. Submit appropriate jobs
70 hold=""
71 for ((i=1;i<=$Npath;i+=1)); do
72     path_in=`echo $paths | cut -d ' ' -f${i}`
73     ori_in=`echo $orientations | cut -d ' ' -f${i}`

```

```

74     Nimgs=`ls ${path_in}/*.png | wc -l`
75     printf 'Orientation #%%d: %s, %s\n' $i $ori_in $path_in
76     qsub -N png2mrc${ori_in} -t 1-${Nimgs} -v
path_in=${path_in},path_out=${path_out},orient=${ori_in} -o ${path_out}/log mpas_png2mrc.q
77     qsub -hold_jid png2mrc${ori_in} -N snr${ori_in} -v
path_in=${path_out}/tmp,path_out=${path_out}/tmp,orient=${ori_in},size="${size}" -o
${path_out}/log mpas_stackAndRotate.q
78     hold=${hold}snr${ori_in},
79     done
80     hold=${hold%?}
81
82     #Average over all orientations
83     qsub -hold_jid $hold -N mpas_avg -v path_in=${path_out}/tmp,path_out=${path_out} -o
${path_out}/log mpas_average.q

```

### C.3.2. mpas\_png2mrc.q

```

1     #! /bin/bash
2
3     #$ -V
4     #$ -cwd
5     #$ -j y
6     #$ -S /bin/bash
7     #$ -m eas
8     #$ -M alexjperez@outlook.com
9     #$ -l h_vmem=1G
10
11     DIR_IMOD=/home/aperez/usr/local/imod_4.8.10/bin
12     DIR_IM=/home/aperez/usr/local/bin
13
14     #Get file to process
15     file_in=`ls ${path_in}/*.png | sed -n "${SGE_TASK_ID}p`
16     base=`basename $file_in`
17     base=${base%.png}
18
19     #Convert PNG to TIF
20     ${DIR_IM}/convert $file_in ${path_out}/tmp/${base}_${orient}.tif
21     ${DIR_IMOD}/tif2mrc ${path_out}/tmp/${base}_${orient}.tif
${path_out}/tmp/${base}_${orient}.mrc
22
23     #Cleanup
24     rm -rf ${path_out}/tmp/${base}_${orient}.tif

```

### C.3.3. mpas\_stackandRotate.q

```

1     #! /bin/bash
2
3     #$ -V
4     #$ -cwd
5     #$ -j y
6     #$ -S /bin/bash
7     #$ -m eas
8     #$ -M alexjperez@outlook.com
9     #$ -l h_vmem=50G

```

```

10
11 DIR_IMOD=/home/aperez/usr/local/imod_4.8.10/bin
12 DIR_IM=/home/aperez/usr/local/bin
13
14 #Stack individual MRCs
15 ${DIR_IMOD}/newstack ${path_in}/*${orient}.mrc ${path_out}/${orient}.st
16 rm -rf ${path_in}/*${orient}.mrc
17
18 size=`echo $size | tr ' ' ','`
19 echo $size
20
21 #Rotate, if necessary
22 if [[ $orient == 'XZ' ]]; then
23     ${DIR_IMOD}/rotatevol -angles 0,0,-90 -size $size ${path_out}/${orient}.st
24     rm -rf ${path_out}/${orient}.st
25 elif [[ $orient == 'YZ' ]]; then
26     ${DIR_IMOD}/rotatevol -angles 0,-90,0 -size $size ${path_out}/${orient}.st
27     rm -rf ${path_out}/${orient}.st
28 fi

```

#### C.3.4. mpas\_average.q

```

1    #! /bin/bash
2
3    # $ -V
4    # $ -cwd
5    # $ -j y
6    # $ -S /bin/bash
7    # $ -m eas
8    # $ -M alexjperez@outlook.com
9    # $ -l h_vmem=50G
10
11   DIR_IMOD=/home/aperez/usr/local/imod_4.8.10/bin
12   DIR_IM=/home/aperez/usr/local/bin
13
14   #Perform averaging
15   ${DIR_IMOD}/clip average ${path_in}/*.st ${path_out}/average.mrc
16
17   #Cleanup
18   #rm -rf ${path_in}/*.st

```

#### C.3.5. msi3d\_dce\_cpd.m

```

1    % msi3d_dce_cpd
2    %   Generates evenly distributed, interpolated binary images between two
3    %   input binary images. This approach is inspired by the morphological
4    %   skeleton interpolation (MSI) algorithm of Chatzis and Pitas [1].
5    %   Depending on the mode specified, the input images are simplified by
6    %   reducing them to their skeleton or perimeter. Distance transform-based
7    %   skeletonization is performed using the discrete curve evolution (DCE)
8    %   algorithm of Bai, et al [2], and perimeterization is performed using
9    %   bwperim. The reduced objects are registered to each other using the

```

```

10 % non-rigid registration mode of the coherent point drift (CPD)
11 % algorithm of Myronenko and Song [3]. CPD determines the non-rigid
12 % mapping of pixels between the input and output reduced objects, and
13 % this correspondence is used to generate evenly spaced, interpolated
14 % objects between the two in the interpolation transformation step.
15 % Finally, the whole objects are reconstructed from the reduced
16 % simplifications. In the case of reduction by skeletonization,
17 % reconstruction is performed by creating the union of all circles
18 % centered at each pixel of the skeleton, with each pixel value
19 % specifying the radius of the given circle. In the case of reduction by
20 % perimeterization, reconstruction is performed by first running a
21 % gap-filling algorithm to connect all pixels of the interpolated
22 % perimeter, then filling the object using imfill(...,'holes').
23 %
24 % Input
25 % -----
26 % I_A,I_B   Input binary images to interpolate between.
27 % L0       Number of interpolated slices to produce between the two
28 %          input images.
29 % mode     = 1, aligns the skeletons of the images. Skeletons are
30 %          generated using the DCE algorithm of Bai, et al.
31 %          = 2, aligns the perimeters of the images.
32 % image    = 1, will write intermediate plots and figures to disk.
33 %          = 0, will not save any intermediate images.
34 % verbose  = 1, will print text pertaining to intermediate steps.
35 %          = 0, will not print any text.
36 % compile  = 1, will compile CPD code. = 0, will not compile,
37 %          assuming code has been previously compiled.
38 %
39 % Output
40 % -----
41 % Out      Stack of interpolated images between I_A and I_B.
42 % time     1x3 vector specifying the runtimes, in seconds, for (1)
43 %          object reduction, (2) CPD registration, and (3)
44 %          interpolation transformation.
45 %
46 % Example
47 % -----
48 % interp = msi3d_cpd_perim( I1,I2,3,3,0,0,1);
49 %
50 % Dependencies
51 % -----
52 % [1] Requires the Coherent Point Drift toolbox, available for download
53 % here: https://sites.google.com/site/myronenko/research/cpd
54 %
55 % [2] Requires the Matlab code for the DCE algorithms for skeleton
56 % generation, available for download here:
57 % https://sites.google.com/site/xiangbai/softwareforskeletonizationandskeletonpru
58 % NOTE: For compatibility with MATLAB R2013a, add the following between
59 % lines 41 and 42 in the file SkeletonGrow1.m:
60 %     lab = single(lab);
61 %
62 % References
63 % -----

```

```

64 % [1] Chatzis and Pitas (2000). Interpolation of 3-D binary images based
65 % on morphological skeletonization. IEEE Transactions on Medical
66 % Imaging, 19(7):699-710.
67 %
68 % [2] Bai, Latecki, and Liu (2007). Skeleton pruning by contour
69 % partitioning with discrete curve evolution. IEEE Transactions on
70 % Pattern Analysis and Machine Intelligence. 29(3):1-14.
71 %
72 % [3] Myronenko and Song (2012). Point Set Registration: Coherent Point
73 % Drift. IEEE Transactions on Pattern Analysis and Machine Intelligence,
74 % 32(12):2262-75.
75 %
76 %
77
78 function [Out,time,ratio] = msi3d_dce_cpd( I_A, I_B, L, mode, image, verbose, compile )
79
80 if nargin < 7; compile = 1; end
81 if nargin < 6; verbose = 0; end
82 if nargin < 5; image = 0; end
83 if nargin < 4; mode = 1; end %Default is to align the skeletons
84
85 % Add needed paths for CPD and compile
86 if compile == 1; compileCPD; end
87
88 % Initialize output matrix
89 Out = zeros([size(I_A),L+2]);
90 Out(:,1) = I_A;
91 Out(:,L+2) = I_B;
92
93 % Set options for CPD
94 opt.method = 'nonrigid';
95 opt.tol = 1e-4;
96 opt.beta = 1;
97 opt.corresp = 1;
98 opt.viz = 0;
99
100 C = 2;
101
102 %%%%%%%%%%%
103 %%% (1) Object Reduction
104 %%%%%%%%%%%
105
106 % Find number of connected components in input images
107 CC_A = bwconncomp(I_A);
108 CC_B = bwconncomp(I_B);
109
110 tic; % Start timer for object reduction
111
112 if mode == 1
113     % Create skeleton images of both input images by using DCE
114     S_A = div_skeleton_new(4,1,~I_A,15);
115     fprintf('Skeletonization of Image A done.\n');
116
117     S_B = div_skeleton_new(4,1,~I_B,15);

```

```

118     fprintf('Skeletonization of Image B done.\n');
119 else
120     S_A = bwperim(I_A);
121     fprintf('Perimeterization of Image A done.\n');
122     S_B = bwperim(I_B);
123     fprintf('Perimeterization of Image B done.\n');
124 end
125
126 runtime_reduce = toc; % End timer for DCE skeletonization
127
128 A_A = numel(find(I_A > 0));
129 A_SA = numel(find(S_A > 0));
130 ratio(1) = A_SA/A_A;
131 fprintf('Image A contains %d points.\n',A_A);
132 fprintf('Reduction A contains %d points.\n',A_SA);
133 fprintf('Reduction : Image Ratio A = %f\n',ratio(1));
134
135 A_B = numel(find(I_B > 0));
136 A_SB = numel(find(S_B > 0));
137 ratio(2) = A_SB/A_B;
138 fprintf('Image B contains %d points.\n',A_B);
139 fprintf('Reduction B contains %d points.\n',A_SB);
140 fprintf('Reduction : Image Ratio B = %f\n',ratio(2));
141
142 %%%%%%%%%%%
143 %%% (2) Skeleton Matching
144 %%%%%%%%%%%
145
146 tic; % Start timer for CPD
147
148 % Convert from image format to a 3xM array of points, as required for CPD
149 S_A_cpd = im2cpd(S_A);
150 S_B_cpd = im2cpd(S_B);
151
152 % Run Coherent Point Drift Algorithm using rigid point set registration.
153 % CPD is run to generate transforms in both directions: (1) From I_A to
154 % I_B, and (2) from I_B to I_A.
155
156 fprintf('Running non-rigid CPD registration of A to B.\n');
157 [Transform_AB,X_AB] = cpd_register(S_B_cpd',S_A_cpd',opt);
158
159 runtime_cpd = toc; % End timer for CPD
160 fprintf('CPD registration done.\n')
161
162 %%%%%%%%%%%
163 % (3) Interpolation Transformation Calculation
164 %%%%%%%%%%%
165
166 % Calculate the transforms in distance, in X and Y, and pixel intensity
167 % in Z to apply to the original skeleton to match the destination
168 % skeleton
169
170 tic; % Start timer for interpolation transformation
171

```

```

172 clear D_AB
173 for i = 1:numel(X_AB); D_AB(:,i) = S_B_cpd(:,X_AB(i)) - S_A_cpd(:,i); end
174
175 for l = 1:L
176
177     % Calculate coefficients for matching interpolations to the properly spaced
178     % slice. l = [1...L].
179     C_AB = l / (L+1);
180     fprintf('Transforming interpolation for l = %d, C_AB = %f.\n',l,C_AB);
181
182     % Scale transforms by the coefficients
183     D_AB_scale = C_AB .* D_AB;
184
185     % Create new objects
186     delta_AB = S_A_cpd + D_AB_scale;
187     delta_AB(1:2,:) = round(delta_AB(1:2,:));
188     S_AB_delta = cpd2im(delta_AB,S_A);
189
190     %%%%%%%%%%%
191     % (4) Object Reconstruction
192     %%%%%%%%%%%
193
194     % Reconstruct objects
195     if mode == 1
196         O_interp_AB = skel2obj(S_AB_delta,2);
197         O_interp_AB = bwmorph(O_interp_AB,'spur');
198         O_interp_AB = bwmorph(O_interp_AB,'hbreak');
199     else
200         O_interp_AB = perimFill(S_AB_delta);
201         %O_interp_AB = imfill(O_interp_AB,'holes');
202     end
203
204     % Check for consistency in connected components. Remove artifacts
205     % if necessary.
206     CC_AB = bwconncomp(O_interp_AB,4);
207
208     if CC_A.NumObjects == CC_B.NumObjects
209         if CC_AB.NumObjects ~= CC_A.NumObjects
210             RP_AB = regionprops(O_interp_AB,'Area','PixelIdxList');
211             [Sort,Idx] = sort([RP_AB.Area],'descend');
212             Remove = Idx(CC_A.NumObjects+1:end);
213             for q = 1:numel(Remove)
214                 O_interp_AB(RP_AB(Remove(q)).PixelIdxList) = 0;
215             end
216         end
217     end
218
219     %%%%%%%%%%%
220     % Store output and set inputs for next iteration
221     %%%%%%%%%%%
222
223     Out(:, :, C) = O_interp_AB;
224     C = C+1;
225

```

```

226     %%%%%%%%%%
227     % (OPTIONAL)
228     %%%%%%%%%%
229
230     if image == 1
231         figure;
232         subplot(3,2,1); imshow(S_A,[]); title('P_A')
233         subplot(3,2,2); imshow(S_B,[]); title('P_B')
234         subplot(3,2,3); imshow(S_interp_AB,[]); title('P_{AB}')
235         subplot(3,2,4); imshow(S_interp_BA,[]); title('P_{BA}')
236         subplot(3,2,5); imshow(O_interp_AB,[]); title('O_{AB}')
237         subplot(3,2,6); imshow(O_interp_BA,[]); title('O_{BA}');
238         figure;
239         subplot(2,3,1); imshow(I_A,[]);
240         subplot(2,3,2); imshow(O_interp_AB,[]);
241         subplot(2,3,3); imshow(I_B,[]);
242         subplot(2,3,4); imshow(I_A,[]);
243         subplot(2,3,5); imshow(O_interp_BA,[]);
244         subplot(2,3,6); imshow(I_B,[]);
245     end
246 end
247
248 runtime_interpTrx = toc; % End timer for interpolation transformation
249
250 Out = uint8(Out);
251
252 fprintf('Run time:\n');
253 fprintf('Image Reduction %f\n',runtime_reduce);
254 fprintf('CPD Registration %f\n',runtime_cpd);
255 fprintf('Interpolation Transformation %f\n',runtime_interpTrx);
256
257 time = [runtime_reduce runtime_cpd runtime_interpTrx];
258
259 end

```

### C.3.6. im2cpd.m

```

1     function Im_cpd = im2cpd( Im )
2     % im2cpd
3     % Converts an image to the 3xM indexed representation needed by the CPD
4     % algorithm.
5     %
6     % Input
7     % -----
8     % Im      Input binary image.
9     %
10    % Output
11    % -----
12    % Im_cpd   Mx3 representation of binary image.
13    %
14    % Example
15    % -----
16    % A_cpd = im2cpd(A);
17

```



```

18     ldx = find( lIm > 0 );
19     lIm_cpd = zeros(3,numel(ldx));
20     [X Y] = ind2sub(size(lIm),ldx);
21     lIm_cpd(1,:) = X';
22     lIm_cpd(2,:) = Y';
23     lIm_cpd(3,:) = lIm(ldx)';
24
25     end

```

### C.3.7. cpd2im.m

```

1     function lIm_out = cpd2im( cpd_mat,lIm )
2     % cpd2im
3     % Converts from the 3xM indexed representation needed by the CPD
4     % algorithm to an image.
5     %
6     % Input
7     % -----
8     % cpd_mat    Input 3xM indexed representation
9     % lIm        Image to match the size of the output image to.
10    %
11    % Output
12    % -----
13    % lIm_out     Output image
14    %
15    % Example
16    % -----
17    % lIm_trx = cpd2im(cpd_trx,lIm);
18
19    [SX SY] = size(lIm);
20    lIm_out = zeros(SX,SY);
21    for i = 1:size(cpd_mat,2)
22        if cpd_mat(1,i) <= 0; cpd_mat(1,i) = 1; end
23        if cpd_mat(2,i) <= 0; cpd_mat(2,i) = 1; end
24        if cpd_mat(3,i) >= 0
25            lIm_out(ceil(cpd_mat(1,i)),ceil(cpd_mat(2,i))) = cpd_mat(3,i);
26        else
27            lIm_out(ceil(cpd_mat(1,i)),ceil(cpd_mat(2,i))) = 0;
28        end
29    end
30    lIm_out = lIm_out(1:SY,1:SY);
31
32    end

```

### C.3.8. skel2obj.m

```

1     % skel2obj
2     % Reconstructs an object from its distance transform-derived skeleton.
3     %
4     % Input
5     % -----
6     % skel       Skeleton image
7     % mode       = 1, uses the midpoint circle algorithm to append
8     %             circles to the image

```



```

61     end
62 end
63
64     end

```

### C.3.9. perimFill.m

```

1     % perimFill
2     % Takes a thinned perimeter binary image, finds the pixels belonging to
3     % gaps in the perimeter, then fills them such that the perimeter can be
4     % properly filled using imfill(...,'holes').
5     %
6     % Input
7     % -----
8     % Im      Binary perimeter image.
9     %
10    % Output
11    % -----
12    % Out     Filled image.
13    %
14    % Example
15    % -----
16    % P_filled = perimFill(P);
17    %
18    % Dependencies
19    % -----
20    % [1] Peter Kovesi's findendsjunctions.m for detecting the pixels
21    % surrounding gaps in the output perimeter map:
22    % http://www.csse.uwa.edu.au/~pk/Research/MatlabFns/#edgelinek
23    % [2] Jing Tian's func_drawLine.m for connecting end pixels:
24    % http://www.mathworks.com/matlabcentral/fileexchange/4211-connect-two-pixels
25    %
26
27    function Out = perimFill( Im )
28
29    % Bridge gaps that are separated by only one pixel
30    Im = bwmorph(Im,'bridge');
31
32    % Find points where ends occur
33    [rj,cj,re,ce] = findendsjunctions(Im,0);
34    re = [re; rj];
35    ce = [ce; cj];
36
37    % Determine the pairs of points in re and ce that constitute both edges of
38    % a gap. This is done by determining which points are closest to one
39    % another by minimizing the distance. The gap is then closed by calling
40    % func_DrawLine.
41    while re
42        R1 = re(1);
43        C1 = ce(1);
44        d = [];
45        for i = 1:numel(re)
46            d(i) = sqrt((R1-re(i))^2 + (C1-ce(i))^2);
47        end

```

```

48     d(1) = Inf;
49     [Min Idx] = min(d);
50     Im = func_DrawLine(Im,R1,C1,re(Idx),ce(Idx),1);
51     re(Idx) = []; re(1) = []; %Remove points from future consideration
52     ce(Idx) = []; ce(1) = [];
53 end
54
55 Out = Im;
56
57 end

```

### C.3.10. msi3d\_display.m

```

1  function msi3d_display( Im,filename,compile )
2  % msi3d_display
3  % Plots a variety of data pertaining to the output interpolations
4  % generated by msi3d_cpd. The plots generated are as follows (from left
5  % to right, top to bottom);
6  %     1. A grayscale overlay of all slices, including the two original
7  %     images and all interpolated images. Each image is coded to a
8  %     unique grayscale value, as displayed in the legend.
9  %     2. A plot of object area versus slice number, as determined by
10 %     regionprops.
11 %     3. A plot of object rotation versus slice number, as determined by
12 %     registry using rigid transformations with CPD.
13 %     4. A plot of the pixel location of the X centroid of each object
14 %     versus slice number, as determined by regionprops.
15 %     5. A plot of the pixel location of the Y centroid of each object
16 %     versus slice number, as determined by regionprops.
17 % If desired, images of the plot will be written to disk in both the
18 % .eps and .tif formats.
19 %
20 % Input
21 % -----
22 % Im      Output image stack from msi3d_cpd.
23 % filename String specifying the filename to save images to. If a
24 %          filename is not specified, images will not be saved to
25 %          disk.
26 % compile = 1, will compile CPD code. = 0, will not compile,
27 %          assuming code has been previously compiled.
28 %
29 % Example
30 % -----
31 % msi3d_display(msi3d_Out);
32 %
33
34 if nargin < 3; compile = 1; end
35 if nargin < 2; filename=""; end
36
37 % Compile CPD code
38 if compile == 1; compileCPD; end
39
40 % Generate pixel values corresponding to each iteration
41 [DimY DimX N] = size(Im);

```

```

42 M = floor(205/N);
43 P = [50+M.*(1:(N-1)) 255];
44 P = fliplr(P);
45
46 % Determine if the image stack is growing or shrinking in size
47 A_start = numel(find(Im(:,:,1) == 1));
48 A_end = numel(find(Im(:,:,N) == 1));
49 if A_start >= A_end
50     j = 1;
51     k = N;
52     inc = 1;
53 else
54     j = N;
55     k = 1;
56     inc = -1;
57 end
58
59 Disp = Im(:,:,1);
60 h = figure;
61 set(h, 'Position', [0 0 1280 800])
62
63 subplot(4,2,7);
64 colormap('Gray');
65
66 for i = j:inc:k
67
68     % Add slices to image at different grayscale values
69     ldx = find( Im(:,:,i) ~= 0 );
70     Disp(ldx) = P(i);
71
72     % Compute statistics on each slice
73     RP = regionprops(Im(:,:,i),'Area','Centroid');
74     A(i) = RP.Area;
75     C(i,:) = RP.Centroid;
76     if i == 1
77         O(i) = 0;
78     else
79         O(i) = imregister_rotational(Im(:,:,1),Im(:,:,i),0);
80     end
81
82     % Create legend patches and text
83     patch([i-1 i-1 i i],[0.75 0.9 0.9 0.75],P(i));
84     if i == 1 | i == N; text(i-1 + 0.5,0.6,num2str(i)); end
85 end
86
87 patch([N N N+1 N+1],[0.75 0.9 0.9 0.75],0);
88 axis([-0.1 N 0 1]);
89 text(0,0.98,'Legend, Z =');
90 axis off; grid off;
91
92 subplot(4,2,[1:2:5]);
93 imshow(Disp,[]);
94
95 X = 1:size(A,2);

```

```

96     DX = (C(:,1) - C(1,1));
97     DY = (C(:,2) - C(1,2));
98
99     % Change interval of X tickmarks if X is too large to display every other
100    % tick properly
101    if N < 20
102        xtick = [1:X(end)];
103    else
104        xtick = [1:2:X(end)];
105    end
106
107    subplot(4,2,2);
108    set(gca, 'FontName', 'Arial');
109    plot(X,A,'ko','LineWidth',4);
110    set(gca,'XTick',xtick);
111    axis([1 X(end) min(A)*0.5 max(A)*1.5]);
112    ylabel('Area (pix)');
113    linearRegression(X,A);
114
115    subplot(4,2,4)
116    plot(X,O,'ko','LineWidth',4);
117    set(gca,'XTick',xtick,'Ytick',-90:45:90);
118    axis([1 X(end) -90 90]);
119    ylabel('Rotation (deg)');
120    linearRegression(X,O);
121
122    subplot(4,2,6)
123    plot(X,DX,'ko','LineWidth',4);
124    set(gca,'XTick',xtick)
125    axis([1 X(end) -DimX DimX]);
126    ylabel('DX Centroid (pix)')
127    linearRegression(X,DX);
128
129    subplot(4,2,8)
130    plot(X,-DY,'ko','LineWidth',4);
131    set(gca,'XTick',xtick)
132    axis([1 X(end) -DimY DimY]);
133    ylabel('DY Centroid (pix)');
134    linearRegression(X,-DY);
135
136    set(findall(h, '-property', 'FontSize'), 'FontSize', 11, 'fontWeight', 'bold')
137
138    % Write file to disk, if desired
139    if ~isempty(filename)
140        options.Format='eps2';
141        hgexport(h,[filename '.eps'],options);
142        options.Format='tiff';
143        hgexport(h,[filename '.tif'],options);
144    end
145
146    %%%%%%%%%%%
%%%%%%%%%%
147

```

```

148 function linearRegression( X,Y )
149     p = polyfit(X,Y,1);
150     yfit = polyval(p,X);
151     yresid = Y - yfit;
152     SSresid = sum(yresid.^2);
153     SStotal = (length(Y)-1)*var(Y);
154     Rsq = 1 - SSresid/SStotal;
155     hold on;
156     p = plot([X(1) X(end)],[yfit(1) yfit(end)],'r-','LineWidth',2);
157     set(p,'Color',[0.25 0.25 0.25]);
158     grid on;
159 end
160
161     %%%%%%%%%%%
%%%%%%%%%%
162
163 end

```

### C.3.11 genCircleTest.m

```

1 function [ I1, I2 ] = genCircleTest(R1,R2,t1,t2)
2 % genCircleTest
3 % Generates a pair of binary images that are circular phantoms of specified
4 % radius and translation from the image center.
5 %
6 % Input
7 % -----
8 % R1,R2      Radii of the circles in pixels
9 % t1,t2      1x2 vectors specifying the X and Y translations for
10 %             the circles. I.E.: t1 = [tx ty].
11 %
12 % Output
13 % -----
14 % I1,I2      Output images
15 %
16 % Example
17 % -----
18 % [I1, I2] = genCircleTest(80,40,[20 10],[0 0])
19 if nargin < 4; t2 = [0 0]; end
20 if nargin < 3; t1 = [0 0]; end
21 if nargin < 2; error('Must specify at least two circle radii.');
```

```

22
23 % Calculate amount to pad each image based on input parameters
24 Max_tx = max(abs([t1 t2]));
25 Max_r = max([R1 R2]);
26 Pad = round((Max_tx + Max_r)/2);
27
28 % Initialize circles
29 I1 = fspecial('disk',R1); I1 = (I1 > 0);
30 I2 = fspecial('disk',R2); I2 = (I2 > 0);
31
32 % Make each image the same size by padding with zeros
33 I1 = padarray(I1,[R2+Pad R2+Pad],0,'both');
```

```

34 I2 = padarray(I2,[R1+Pad R1+Pad],0,'both');
35
36 % Translate circles
37 I1 = circTranslate(I1,t1(1),t1(2));
38 I2 = circTranslate(I2,t2(1),t2(2));
39
40
41 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
42 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
43 function Im_out = circTranslate( Im_in, dC, dR )
44     tx = maketform('affine',[1 0; 0 1; dC dR]);
45     [SX SY] = size(Im_in);
46     bounds = findbounds(tx,[1 1; size(Im_in)]);
47     bounds(1,:) = [1 1];
48     Im_out = imtransform(Im_in,tx,'XData',bounds(:,2),'YData',bounds(:,1));
49     Min = min([dC dR]);
50     if Min < 0
51         Im_out = padarray(Im_out,[abs(Min) abs(Min)],0,'post');
52     end
53     Im_out = Im_out(1:SY,1:SY);
54 end
55
56 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
57 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
58 end

```

### C.3.12. genSquareTest.m

```

1 function [ I1, I2 ] = genSquareTest(S1,S2,t1,t2,ang1,ang2)
2 % genSquareTest
3 % Generates a pair of binary images that are square phantoms of
4 % specified size, XY rotation, and translation from the image center.
5 %
6 % Input
7 % -----
8 % S1,S2 Length of the squares in pixels.
9 % t1,t2 1x2 vectors specifying the X and Y translations for
10 % the squares. I.E.: t1 = [tx ty].
11 % ang1,ang2 Angle, in degrees, to rotate the squares about the XY
12 % axis.
13 %
14 % Output
15 % -----
16 % I1,I2 Output images
17 %
18 % Example
19 % -----
20 % [I1, I2] = genSquareTest(80,40,[20 10],[0 0],0,30)
21 %
22 % Dependencies
23 % -----

```



```

24 % Requires Jan Motl's rotateAround.m from the Mathworks File Exchange:
25 % http://www.mathworks.com/matlabcentral/fileexchange/40469-rotate-an-image-
about-a-point
26
27 if nargin < 6; ang2 = 0; end
28 if nargin < 5; ang1 = 0; end
29 if nargin < 4; t2 = [0 0]; end
30 if nargin < 3; t1 = [0 0]; end
31 if nargin < 2; error('Must specify at least two square sizes.');
```

end

```

32
33 % Calculate amount to pad each image based on input parameters
34 Max_tx = max(abs([t1 t2]));
35 Max_r = max([S1 S2]);
36 Pad = round((Max_tx + Max_r));
37
38 % Initialize squares
39 I1 = ones(S1,S1);
40 I2 = ones(S2,S2);
41
42 % Make each image the same size by padding with zeros
43 Max = abs(max([S1 S2]));
44
45 I1 = padarray(I1,[ceil((Max - S1)/2)+Pad ceil((Max - S1)/2)+Pad],0,'both');
46 I2 = padarray(I2,[ceil((Max - S2)/2)+Pad ceil((Max - S2)/2)+Pad],0,'both');
47 if mod((Max - S1),2) ~= 0; I2 = padarray(I2,[1 1],0,'post'); end
48 if mod((Max - S2),2) ~= 0; I1 = padarray(I1,[1 1],0,'post'); end
49
50 % Translate squares
51 I1 = sqTranslate(I1,t1(1),t1(2));
52 I2 = sqTranslate(I2,t2(1),t2(2));
53
54 % Rotate squares. Rotation is performed about the centroid of each square
55 % using rotateAround.m.
56 RP_I1 = regionprops(I1,'Centroid');
57 RP_I2 = regionprops(I2,'Centroid');
58 I1 = rotateAround(I1,RP_I1.Centroid(2),RP_I1.Centroid(1),ang1);
59 I2 = rotateAround(I2,RP_I2.Centroid(2),RP_I2.Centroid(1),ang2);
60
61
62 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
63 function Im_out = sqTranslate( Im_in, dC, dR )
64     tx = maketform('affine',[1 0; 0 1; dC dR]);
65     [SX SY] = size(Im_in);
66     bounds = findbounds(tx,[1 1; size(Im_in)]);
67     bounds(1,:) = [1 1];
68     Im_out = imtransform(Im_in,tx,'XData',bounds(:,2),'YData',bounds(:,1));
69     Min = min([dC dR]);
70     if Min < 0
71         Im_out = padarray(Im_out,[abs(Min) abs(Min)],0,'post');
72     end
73     Im_out = Im_out(1:SY,1:SY);
74 end

```

```

75
76     %%%%%%%%%%
%%%%%%%%%
77
78     end

```

### C.3.13. genArbitraryTest.m

```

1     function [I1,I2] = genArbitraryTest( Im_string,scale1,scale2,t1,t2,ang1,ang2 );
2     % genArbitraryTest
3     % Takes a binary image as input, then creates two phantoms of this image
4     % that are scaled, translated, and rotated by the specified parameters.
5     %
6     % Input
7     % -----
8     % Im_string    String specifying the path of the binary image to be
9     %               loaded and made into phantoms.
10    % scale1,scale2  Factor by which to scale the two phantoms with respect
11    %               to the input image.
12    % t1,t2         1x2 vectors specifying the X and Y translations for
13    %               the phantoms. I.E.: t1 = [tx ty].
14    % ang1,ang2     Angle, in degrees, to rotate the phantoms about the XY
15    %               axis.
16    %
17    % Output
18    % -----
19    % I1,I2        Output images
20    %
21    % Example
22    % -----
23    % [I1, I2] = genArbitraryTest('nucleus.0100.tif',1,0.5,[200 100],[0 0],0,30)
24    %
25    % Dependencies
26    % -----
27    % Requires Jan Motl's rotateAround.m from the Mathworks File Exchange:
28    % http://www.mathworks.com/matlabcentral/fileexchange/40469-rotate-an-image-
29    % about-a-point
30    %
31    % if nargin < 7; ang2 = 0; end
32    % if nargin < 6; ang1 = 0; end
33    % if nargin < 5; t2 = [0 0]; end
34    % if nargin < 4; t1 = [0 0]; end
35    % if nargin < 3; scale2 = 1; end
36    % if nargin < 2; scale1 = 1; end
37    %
38    %
39    % Crop input image to be tight around the binary blob
40    % RP = regionprops(Im,'BoundingBox');
41    % BB = RP.BoundingBox;
42    % BB(1:2) = floor(BB(1:2));
43    % BB(3:4) = ceil(BB(3:4));
44    % Im = Im(BB(2):BB(2)+BB(4),BB(1):BB(1)+BB(3));

```

```

45
46 % Scale image
47 I1 = imresize(I1,scale1);
48 I2 = imresize(I2,scale2);
49
50 [Y1 X1] = size(I1);
51 [Y2 X2] = size(I2);
52 [MaxY IdxMaxY] = max([Y1 Y2]);
53 [MaxX] = max([X1 X2]);
54
55 % Pad the smaller image to be the same size as the larger image,
56 % post-scaling.
57 if scale1 ~= scale2 & IdxMaxY == 1
58     I2 = padarray(I2,[ceil((MaxY-Y2)/2) ceil((MaxX-X2)/2)],0,'both');
59     if mod((MaxY-Y2),2) == 1; I1 = padarray(I1,[1 0],0,'post'); end
60     if mod((MaxX-X2),2) == 1; I1 = padarray(I1,[0 1],0,'post'); end
61 elseif scale1 ~= scale2 & IdxMaxY == 2
62     I1 = padarray(I1,[ceil((MaxY-Y1)/2) ceil((MaxX-X1)/2)],0,'both');
63     if mod((MaxY-Y1),2) == 1; I2 = padarray(I2,[1 0],0,'post'); end
64     if mod((MaxX-X1),2) == 1; I2 = padarray(I2,[0 1],0,'post'); end
65 end
66
67 % Pad again to account for translations
68 Max = max([t1 t2]);
69 I1 = padarray(I1,[Max+round(MaxY/2) Max+round(MaxX/2)],0,'both');
70 I2 = padarray(I2,[Max+round(MaxY/2) Max+round(MaxX/2)],0,'both');
71
72 % Translate
73 I1 = arbTranslate(I1,t1(1),t1(2));
74 I2 = arbTranslate(I2,t2(1),t2(2));
75
76 % Rotate
77 RP_I1 = regionprops(I1,'Centroid');
78 RP_I2 = regionprops(I2,'Centroid');
79 I1 = rotateAround(I1,RP_I1.Centroid(2),RP_I1.Centroid(1),ang1);
80 I2 = rotateAround(I2,RP_I2.Centroid(2),RP_I2.Centroid(1),ang2);
81
82 figure; imshow(I1,[])
83 figure; imshow(I2,[])
84
85
86 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
87 function Im_out = arbTranslate( Im_in, dC, dR )
88     tx = maketform('affine',[1 0; 0 1; dC dR]);
89     [SX SY] = size(Im_in);
90     bounds = findbounds(tx,[1 1; size(Im_in)]);
91     bounds(1,:) = [1 1];
92     Im_out = imtransform(Im_in,tx,'XData',bounds(:,2),'YData',bounds(:,1));
93     Min = min([dC dR]);
94     if Min < 0
95         Im_out = padarray(Im_out,[abs(Min) abs(Min)],0,'post');
96     end

```

```

97     Im_out = Im_out(1: SX, 1: SY);
98     end
99
100    %%%%%%%%%%%
%%%%%%%%%%
101
102     end

```

### C.3.14. contourgen.sh

```

1     #! /bin/bash
2
3     function show_help () {
4     cat <<-END
5     contourgen.sh
6     Usage:
7     -----
8     -i | --input (Directory name)
9         Path to series of TIF files to be processed
10
11     -o | --output (Directory name)
12         Path to store output model file to
13
14     -m | --mrc (MRC stack)
15         Name of MRC stack to match to
16
17     -d | --del (Integer,Integer,Integer)
18         Pixel size of MRC stack to match to (X,Y,Z)
19
20     -r | --org (Integer,Integer,Integer)
21         Origin of MRC stack to match to (X,Y,Z)
22
23     -R | --point
24         Tolerance for point shaving during model generation.
25         R = [0,...,1]. Default value = 0.
26
27     -k | --sigma
28         Smooth the data during model generation with a kernel filter
29         whose Gaussian sigma is given by this value. Default value = 0.
30
31     -h | --help
32         Display this help
33     END
34     }
35
36     while ;; do
37         case $1 in
38             -h|--help)
39                 show_help
40                 exit
41                 ;;
42             -i|--input)
43                 path_in=$2

```

```

44         shift 2
45         continue
46         ;;
47     -o|--output)
48         path_out=$2
49         shift 2
50         continue
51         ;;
52     -m|--mrc)
53         mrc_stack=$2
54         shift 2
55         continue
56         ;;
57     -d|--del)
58         del=$2
59         shift 2
60         continue
61         ;;
62     -r|--org)
63         org=$2
64         shift 2
65         continue
66         ;;
67     -R|--point)
68         pointred=$2
69         shift 2
70         continue
71         ;;
72     -k|--sigma)
73         sigma=$2
74         shift 2
75         continue
76         ;;
77     *)
78         break
79     esac
80     shift
81 done
82
83 #Check for problems with input. Print help and exit if not correct
84 if [[ ! $path_in ]] || [[ ! $path_out ]]; then
85     printf 'ERROR: options -i and -o must be specified\n\n' >&2
86     show_help
87     exit 1
88 fi
89
90 if [[ -n $mrc_stack ]] && [[ -n $del ]]; then
91     printf 'ERROR: Use either the -m option OR the -d and -r options\n\n' >&2
92     show_help
93     exit 1
94 elif [[ -n $mrc_stack ]] && [[ -n $org ]]; then
95     printf 'ERROR: Use either the -m option OR the -d and -r options\n\n' >&2
96     show_help
97     exit 1

```

```

98     elif [[ ! $mrc_stack ]] && [[ ! $del ]] && [[ -n $org ]]; then
99         printf 'ERROR: options -d and -r must both be specified\n\n' >&2
100        show_help
101        exit 1
102     elif [[ ! $mrc_stack ]] && [[ -n $del ]] && [[ ! $org ]]; then
103         printf 'ERROR: options -d and -r must both be specified\n\n' >&2
104        show_help
105        exit 1
106     fi
107
108     source /home/aperez/.bashrc #Source IMOD
109
110     #Make output directory if necessary and make temporary subdirectories
111     if [[ ! -d $path_out ]]; then mkdir ${path_out}; fi
112     mkdir ${path_out}/log ${path_out}/mod ${path_out}/ncont ${path_out}/txt
113
114     Nslices=`ls ${path_in}/*.tif | wc -l` #Determine number of images
115
116     #If the original mrc stack is supplied, extract the pixel spacing and origin information
117     #from the header of that file. If not, use the user-supplied values. These values are
118     #critical to ensure the output model file aligns properly with the original mrc stack.
119     if [[ -n $mrc_stack ]]; then
120         del=`${IMOD_DIR}/bin/header -pixel $mrc_stack | tr -s '`
121         org=`${IMOD_DIR}/bin/header -origin $mrc_stack | tr -s '`
122     else
123         del=`echo $del | tr ',' '` #Replace commas with spaces
124         org=`echo $org | tr ',' '`
125     fi
126
127     #Turn off point reduction and smoothing (i.e., set their values to zero) if they are not
specified
128     if [[ ! $pointred ]]; then pointred=0; fi
129     if [[ ! $sigma ]]; then sigma=0; fi
130
131     #(1) Submit tif2mod2D.q as an array job to convert TIFS to model files.
132     qsub -t 1-${Nslices} -v
path_in=${path_in},path_out=${path_out},del="${del}",org="${org}",pointred=${pointred},sigma=${
sigma} -o ${path_out}/log tif2mod2D.q
133
134     #(2) Submit mod2point2D.q as an array job to convert model files to text files containing
point listings.
135     qsub -hold_jid tif2mod -t 1-${Nslices} -v
path_mod=${path_out}/mod,path_txt=${path_out}/ncont,path_out=${path_out}/txt -o
${path_out}/log mod2point2D.q
136
137     #(3) Submit point2mod3D.q to append all point listings to a single text file, and then
generate a model from this using point2model.
138     qsub -hold_jid mod2point -v path_out=${path_out},del="${del}",org="${org}" -o
${path_out}/log point2mod3D.q

```

### C.3.15. tif2mod2D.q

```

1     #! /bin/bash
2

```

```

3      # tif2mod2D.q
4      #   SGE script to convert a single binary image to a single model file. The output model
file
5      #   consists of a single object with individual contours drawn around each 2D connected
component.
6      #   The steps involved are:
7      #       (1) Convert the TIF file to a 2D mrc file
8      #       (2) Alter the header information of the 2D mrc file to match that of the original stack
9      #       (3) Generate contours around 2d connected components using user-specified
values for
10     #           point reduction (-R) and Gaussian smoothing (-k).
11     #       (4) Translate the model file in Z so it sits in the correct depth of the stack.
12     #       (5) Convert the IMOD model file binary format to ASCII, and parse this output to
determine
13     #           the number of contours on this slice. Store this value to a text file, which will be
14     #           used in future processing.
15     #       (6) Clean up intermediates.
16     #
17     #   The next script to be run in the workflow is mod2point2D.q
18     #
19     #   INPUT
20     #   -----
21     #   path_in   Path containing the segmented TIF images to process
22     #   path_out  Output path to write model files to
23     #   del       Pixel spacing of the original mrc file, delimited by spaces, in the format "X Y
Z"
24     #   org       Origin of the original mrc file, delimited by spaces, in the format "X Y Z"
25     #   pointred  Value for point reduction during contour generation
26     #   sigma     Value for Gaussian smoothing during contour generation
27     #
28
29     # $ -S /bin/bash
30     # $ -N tif2mod
31     # $ -j yes
32     # $ -m eas
33     # $ -M alexjperez@outlook.com
34     # $ -l h_vmem=1G
35     # $ -cwd
36     # $ -V
37
38     source /home/aperez/.bashrc #Source IMOD
39
40     file_in=`ls ${path_in}/*.tif | sed -n "${SGE_TASK_ID}p" #Determine which image to work
with
41     base=`basename $file_in`
42     base=${base%.*}
43
44     #STEP (1)
45     ${IMOD_DIR}/bin/tif2mrc $file_in ${path_out}/mod/${base}.mrc
46
47     #STEP (2)
48     echo -e "${path_out}/mod/${base}.mrc\ndel\n${del}\norg\n${org}\ndone\n" |
${IMOD_DIR}/bin/alterheader
49

```

```

50 #STEP (3)
51 ${IMOD_DIR}/bin/imodauto -h 1 -R $pointred -k $sigma ${path_out}/mod/${base}.mrc
${path_out}/mod/${base}.mod
52
53 #STEP (4)
54 ${IMOD_DIR}/bin/imodtrans -tz $((SGE_TASK_ID-1)) ${path_out}/mod/${base}.mod
${path_out}/mod/${base}.mod
55
56 #STEP (5)
57 ${IMOD_DIR}/bin/imodinfo -a ${path_out}/mod/${base}.mod | grep -m 1 'object 0*' | cut -d
'' -f3 >> ${path_out}/ncont/${base}.txt
58
59 #STEP (6)
60 rm -rf ${path_out}/mod/${base}.mrc ${path_out}/mod/${base}.mod~

```

### C.3.16. mod2point2D.q

```

1 #! /bin/bash
2
3 # mod2point2D.q
4 # SGE script to submit array jobs converting 2D model files to point listings compatible
5 # with the IMOD program point2model. Conversion is done using MATLAB, and is
dependent on
6 # the MatTomo package of MATLAB scripts from IMOD.
7 #
8 # INPUT
9 # -----
10 # path_mod Path containing the model files to process
11 # path_txt Path containing the contour listings output from tif2mod2D.q
12 # path_out Output path to write point listing text files to
13 #
14
15 # $ -S /bin/bash
16 # $ -N mod2point
17 # $ -j yes
18 # $ -m eas
19 # $ -M alexjperez@outlook.com
20 # $ -l h_vmem=5G
21 # $ -cwd
22 # $ -V
24 matlab -nodisplay -nosplash -r
"mod2point2D("${path_mod}","${path_txt}","${path_out}","${SGE_TASK_ID}");

```

### C.3.17. mod2point2D.m

```

1 function mod2point2D( path_models, path_txts, path_out, N )
2 % Converts an IMOD model file to a text file where each line corresponds to
3 % one point of the model file. Each line consists of five numbers, arranged
4 % as such: Object, Contour, X, Y, Z
5 %
6 % The starting contour is determined by parsing the text files output from
7 % tif2mod2D.q.
8 %
9 % INPUT

```



```

10 % -----
11 % path_models Path containing the input IMOD model files
12 % path_txts Path containing the text files of contour listings
13 % generated by tif2mod2D.q
14 % path_out Output path to write text files to
15 % N Model number within path_models to process
16 %
17 % DEPENDENCIES
18 % -----
19 % The MatTomo package of PEET is needed:
20 % http://bio3d.colorado.edu/imod/matlab.html
21 %
22 % EXAMPLE
23 % -----
24 % mod2point2D('out/mod','out/ncont','out/txt',100);
25 %
26
27 path_mattomo = '/data/aperez/mfiles/MatTomo'; %Path containing MatTomo
28 addpath(genpath(path_mattomo));
29
30 tic;
31 models = dir(fullfile(path_models, '*.mod'));
32 txts = dir(fullfile(path_txts, '*.txt'));
33
34 %Determine the starting contour number by reading and summing all contour
35 %listings before the current value of N
36 C = 1;
37 for i = 1:N-1
38     C = C + load(fullfile(path_txts,txts(i).name));
39 end
40
41 filei = fullfile(path_models,models(N).name); %Name of model file
42 modi = ImodModel(filei); %Read model file to MATLAB
43 obji = getObject(modi,1); %Get first object (should be the only object)
44 Ncont = getNContours(obji); %Determine the total number of contours
45 text_out = fullfile(path_out,['test_' sprintf('%03d',N) '.txt']);
46 fprintf('Processing model %s\n',filei);
47 fprintf('Model %s has %d contours.\n',filei,Ncont);
48 for j = 1:Ncont %Loop over all contours
49     contj = getContour(obji,j); %Read contour
50     points = getPoints(contj); %Extract contour's points to a matrix
51     [NR,NC] = size(points);
52     points2write = zeros(NR,NC+2); %Generate a new matrix to add obj and cont listings
53     points2write(:,1) = 1; %All contours will belong to object 1
54     points2write(:,2) = C; %Second value is the contour number
55     points2write(:,3:5) = points;
56     %Append to the text file. The space delimiter is required by point2model. A high
57     %precision is required because if the contour number exceeds this number of digits,
58     %dmlwrite will output it in scientific notation, which point2model cannot understand.
59     %A precision of 9 means the file can have one billion contours before this limit is
60     %exceeded.
61     dlmwrite(text_out,points2write,'delimiter',' ','-append','precision',9);
62     C = C + 1;
63 end

```

```

64
65     fprintf('Elapsed time: %0.2f seconds\n',toc);
66
67     end

```

### C.3.18. point2mod3D.q

```

1     #! /bin/bash
2
3     # point2mod3D.q
4     # Takes a directory of text files containing point listings for each 2D slice, appends
5     # them to one single file, and generates contours using this file as input to the IMOD
6     # program point2model.
7     #
8     # INPUT
9     # -----
10    # path_out  Output path
11    # del      Pixel spacing of the original mrc file, delimited by spaces, in the format "X Y
Z"
12    # org      Origin of the original mrc file, delimited by spaces, in the format "X Y Z"
13    #
14
15    # $ -S /bin/bash
16    # $ -N point2mod
17    # $ -j yes
18    # $ -m eas
19    # $ -M alexjperez@outlook.com
20    # $ -l h_vmem=5G
21    # $ -cwd
22    # $ -V
23
24    source /home/aperez/.bashrc #Source IMOD
25
26    del=`echo $del | tr -s ' ','` #Replace space delimiter with commas
27    org=`echo $org | tr -s ' ','`
28
29    rm -rf ${path_out}/mod ${path_out}/ncont #Remove intermediates
30
31    for file in ${path_out}/txt/*.txt; do #Append individual point listing files to one file
32        cat $file >> ${path_out}/out.txt
33    done
34
35    #Generate a model file from the complete point listing
36    point2model -pixel ${del} -origin ${org} ${path_out}/out.txt ${path_out}/out.mod
37
38    rm -rf ${path_out}/out.txt ${path_out}/txt

```

### C.3.19. sbem\_analyze\_nuclei.sh

```

1     #! /bin/bash
2
3     function show_help () {
4         cat <<-END
5

```

```

6  sbem_analyze_nuclei.sh
7  Usage:
8  -----
9  -h | --help
10     Display this help
11  -i | --input
12     IMOD model file containing objects of interest
13  -c1
14     Color string for nuclei, in the format R,G,B (i.e., 1,1,0)
15  -c2
16     Color string for nucleoli, in the format R,G,B (i.e., 0,1,1)
17  Example:
18  -----
19  ./sbem_analyze_nuclei_nucleoli.sh -i ZT04_01_join.mod -c1 1,1,0 -c2 0,0,1
20
21  END
22  }
23
24  function scientific_to_bc () {
25     base=`echo ${1} | cut -d 'e' -f1 | bc`
26     exp=`echo ${1} | cut -d '+' -f2 | bc`
27     echo "${base}*10^{exp}"
28  }
29
30  #####
31  ## (0) Parse input arguments
32  #####
33
34  while ;; do
35     case $1 in
36         -h|--help)
37             show_help
38             exit
39             ;;
40         -i|--input)
41             file=$2
42             shift 2
43             continue
44             ;;
45         -c1)
46             c1=$2
47             shift 2
48             continue
49             ;;
50         -c2)
51             c2=$2
52             shift 2
53             continue
54             ;;
55         *)
56             break
57     esac
58     shift
59 done

```

```

60
61 if [ ! "$file" ] | [ ! "$c1" ] | [ ! "$c2" ]; then
62     echo 'ERROR: options -i, -c1, and -c2 must be specified. See -help' >&2
63     exit 1
64 fi
65
66 #####
67 ## (1) Group nuclei with their corresponding nucleoli. This is done by taking the bounding
box of each nucleus and
68 ## finding the centroids of nucleoli that lie within this bounding box.
69 #####
70
71 #Determine zscale of the model
72 zscale=`imodinfo -a $file | grep -m 1 'scale' | cut -d ' ' -f4`
73
74 #Decompose color strings
75 for i in 1 2 3; do
76     cnuc[${i}]=`echo $c1 | cut -d ',' -f${i}`
77     cnucl[${i}]=`echo $c2 | cut -d ',' -f${i}`
78 done
79
80 #Parse IMOD model file for objects matching nuclear color and separate them into a new
model file
81 imodinfo -a $file | grep -B2 "color ${cnuc[1]} ${cnuc[2]} ${cnuc[3]}" | awk -F '\n' 'ln ~ /^$/ {
ln = "matched"; print $1 } $1 ~ /^--$/ { ln = "" }' | cut -d ' ' -f2 >> sann_nuc.txt
82 Nnuc=`cat sann_nuc.txt | wc -l`
83 str_nuc=""
84 while read line; do
85     str_nuc=${str_nuc}${str_nuc}$(line+1)),
86 done < sann_nuc.txt
87 str_nuc=${str_nuc%??}
88 rm -rf sann_nuc.txt
89 imodextract $str_nuc $file ${file}_nuclei
90
91 #Parse IMOD model file for objects matching nucleolar color and separate them into a
new model file
92 imodinfo -a $file | grep -B2 "color ${cnucl[1]} ${cnucl[2]} ${cnucl[3]}" | awk -F '\n' 'ln ~ /^$/ {
ln = "matched"; print $1 } $1 ~ /^--$/ { ln = "" }' | cut -d ' ' -f2 >> sann_nucl.txt
93 Nnucl=`cat sann_nucl.txt | wc -l`
94 str_nucl=""
95 while read line; do
96     str_nucl=${str_nucl}${str_nucl}$(line+1)),
97 done < sann_nucl.txt
98 str_nucl=${str_nucl%??}
99 rm -rf sann_nucl.txt
100 imodextract $str_nucl $file ${file}_nucleoli
101
102 #Create a temporary text file with the centroids of each nucleolus
103 for ((i=1;i<=${Nnucl};i+=1)); do
104     strcent=`imodinfo -o $i -F ${file}_nucleoli | grep 'Center' | tr -s ' ' | cut -d '(' -f2 | cut -d ')'
-f1`
105     xcent=`echo $strcent | cut -d ',' -f1`
106     ycent=`echo $strcent | cut -d ',' -f2`
107     zcent=`echo $strcent | cut -d ',' -f3`

```

```

108     zcent=`echo "$zcent / $zscale" | bc`
109     xcent=`echo "($xcent+0.5)/1" | bc`
110     ycent=`echo "($ycent+0.5)/1" | bc`
111     zcent=`echo "($zcent+0.5)/1" | bc`
112     echo $xcent $ycent $zcent >> sann_cent.txt
113 done
114
115 #Find nucleoli that belong to each nucleus
116 for ((i=1;i<=${Nnuc};i+=1)); do
117     strbb=`imodinfo -o $i -F ${file}_nuclei | grep 'Bounding Box' | cut -d '{' -f2 | cut -d '}' -f1`
118     strcent=`imodinfo -o $i -F ${file}_nuclei | grep 'Center' | tr -s ' ' | cut -d '(' -f2 | cut -d ')' -f1`
119     xcentnuc=`echo $strcent | cut -d ',' -f1`
120     ycentnuc=`echo $strcent | cut -d ',' -f2`
121     zcentnuc=`echo $strcent | cut -d ',' -f3`
122     zcentnuc=`echo "$zcentnuc / $zscale" | bc`
123     xmin=`echo $strbb | cut -d ',' -f1 | cut -d '(' -f2`
124     ymin=`echo $strbb | cut -d ',' -f2`
125     zmin=`echo $strbb | cut -d ',' -f3 | cut -d ')' -f1`
126     xmax=`echo $strbb | cut -d ',' -f4 | cut -d '(' -f2`
127     ymax=`echo $strbb | cut -d ',' -f5`
128     zmax=`echo $strbb | cut -d ',' -f6 | cut -d ')' -f1`
129     xcentnuc=`echo "($xcentnuc+0.5)/1" | bc`
130     ycentnuc=`echo "($ycentnuc+0.5)/1" | bc`
131     zcentnuc=`echo "($zcentnuc+0.5)/1" | bc`
132     xmin=`echo "($xmin+0.5)/1" | bc`
133     ymin=`echo "($ymin+0.5)/1" | bc`
134     zmin=`echo "($zmin+0.5)/1" | bc`
135     xmax=`echo "($xmax+0.5)/1" | bc`
136     ymax=`echo "($ymax+0.5)/1" | bc`
137     zmax=`echo "($zmax+0.5)/1" | bc`
138     printf "Nucleus #%d: %d,%d,%d %d,%d,%d\n" $i $xmin $ymin $zmin $xmax $ymax
139     $zmax
140     strkeep=""
141     count=1
142     for ((j=1;j<=${Nnucl};j+=1)); do
143         xcent=`sed "$j;q;d' sann_cent.txt | cut -d ' ' -f1`
144         ycent=`sed "$j;q;d' sann_cent.txt | cut -d ' ' -f2`
145         zcent=`sed "$j;q;d' sann_cent.txt | cut -d ' ' -f3`
146         if (($xcent < $xmax)) && (($xcent > $xmin)) && (($ycent < $ymax)) && (($ycent > $ymin)) && (($zcent < $zmax)) && (($zcent > $zmin)); then
147             printf " Nucleolus #%d: %d,%d,%d\n" $j $xcent $ycent $zcent
148             echo $count $xcentnuc $ycentnuc $zcentnuc >> sann_cent_nucl.txt
149             echo $count $xcent $ycent $zcent >> sann_cent_nucl.txt
150             strkeep=${strkeep}$j},
151             count=$((count+1))
152         fi
153     done
154     strkeep=${strkeep%?}
155     filei=nucleus_`printf "%03d" $i`.mod
156     imodextract $i ${file}_nuclei tempnucleus.mod
157     imodextract $strkeep ${file}_nucleoli tempnucleoli.mod
158     point2model -input sann_cent_nucl.txt -output sann_cent_nucl.mod -circle 6 -sphere
159     20 -color 255,0,0 > /dev/null

```

```

158     imodjoin tempnucleus.mod tempnucleoli.mod sann_cent_nucl.mod $filei > /dev/null
159     rm -rf tempnucleus.mod tempnucleoli.mod sann_cent_nucl*
160 done
161 rm -rf sann_cent.txt
162
163 #####
164 ## (2) Volume and surface area measurements
165 #####
166
167 for ((i=1;i<=${Nnuc};i+=1)); do
168     filei=nucleus_`printf "%03d" $i`.mod
169     Nobj=`imodinfo -a $filei | grep -m 1 '^imod' | cut -d ' ' -f2`
170     echo `${Nobj-2}` >> sann_sort.txt
171 done
172 maxnucl=`sort -r sann_sort.txt | head -1`
173
174 for ((i=1;i<=${Nnuc};i+=1)); do
175     printf "Analyzing morphology for Nucleus #`d\n" $i
176     filei=nucleus_`printf "%03d" $i`.mod
177     Nobj=`imodinfo -a $filei | grep -m 1 '^imod' | cut -d ' ' -f2`
178     imodmesh -e $filei $filei > /dev/null
179     imodmesh -C -T -P 100 -I $filei $filei > /dev/null
180     strvol=`imodinfo -c -o 1 $filei | grep "#--" -A1 | tail -1 | tr -s ' '`
181     volnuc=`echo $strvol | cut -d ' ' -f4`
182     sanuc=`echo $strvol | cut -d ' ' -f5`
183     volnuc_bc=`scientific_to_bc $volnuc`
184     sanuc_bc=`scientific_to_bc $sanuc`
185     volnuc_um=`echo "scale=4; $volnuc_bc / 1000^3" | bc`
186     sanuc_um=`echo "scale=4; $sanuc_bc / 1000^2" | bc`
187     savratio=`echo "scale=4; $sanuc_um / $volnuc_um" | bc`
188     echo $volnuc_um $sanuc_um $savratio >> nucleus_morphology.txt
189     printf "%d " `${Nobj-2}` >> nucleolus_morphology.txt
190     sumvolnucl=0
191     for ((j=2;j<=${maxnucl+1};j+=1)); do
192         if (( $j <= `${Nobj-1}` )); then
193             strvol=`imodinfo -c -o $j $filei | grep "#--" -A1 | tail -1 | tr -s ' '`
194             volnuclj=`echo $strvol | cut -d ' ' -f4`
195             volnuclj_bc=`scientific_to_bc $volnuclj`
196             volnuclj_um=`echo "scale=4; $volnuclj_bc / 1000^3" | bc`
197             sumvolnucl=`echo "$sumvolnucl + $volnuclj_um" | bc`
198             printf "%0.4f " $volnuclj_um >> nucleolus_morphology.txt
199         else
200             printf "0 " >> nucleolus_morphology.txt
201         fi
202     done
203     nuclvolfrac=`echo "scale=4; $sumvolnucl / $volnuc_um" | bc`
204     printf "%0.6f %0.4f\n" $sumvolnucl $nuclvolfrac >> nucleolus_morphology.txt
205 done
206
207 #####
208 ## (3) Distance to centroid measurements
209 #####
210
211 for ((i=1;i<=${Nnuc};i+=1)); do

```

```

212     printf "Analyzing centroid distances for Nucleus #%"$i\n" $i
213     filei=nucleus_`printf "%03d" $i`.mod
214     Nobj=`imodinfo -a $filei | grep -m 1 '^imod' | cut -d ' ' -f2`
215     Nnucli=$((Nobj-2))
216     for ((j=1;j<=${Nnucli};j+=1)); do
217         dist=`imodinfo -l -o $Nobj $filei | grep '#--' -A${Nnucli} | sed -n "$((j+1))p" | tr -s ' ' |
cut -d ' ' -f5`
218         dist_um=`echo "scale=4; $dist / 1000" | bc`
219         printf "%0.4f " $dist_um >> dist_centroid.txt
220     done
221     printf "\n" >> dist_centroid.txt
222 done
223
224 #####
225 ## (4) Distance to nuclear envelope measurements
226 #####
227
228 for ((i=1;i<=${Nnuc};i+=1)); do
229     printf "Analyzing nuclear envelope distances for Nucleus #%"$i\n" $i
230     filei=nucleus_`printf "%03d" $i`.mod
231     Nobj=`imodinfo -a $filei | grep -m 1 '^imod' | cut -d ' ' -f2`
232     Nnucli=$((Nobj-2))
233     strobj=`seq -s ',' 2 ${Nnucli+1}`
234
strmtk="\n\n1\n0\n${filei}\n\n\n0,0\n0.05,200\n0\n2,0\n1\n0\n1\n${strobj}\n1\n17\n0,5\n0.5,20\n0\n
0\n21\n${filei}\n\n25\n"
235     echo $Nobj
236     echo $Nnucli
237     echo $strobj
238     echo $strmtk
239     echo -e $strmtk | mtk > /dev/null
240     for ((j=1;j<=${Nnucli};j+=1)); do
241         dist=`imodinfo -l -o ${Nobj+1} $filei | grep '#--' -A${Nnucli} | sed -n "$((j+1))p" | tr -s
' ' | cut -d ' ' -f5`
242         dist_um=`echo "scale=4; $dist / 1000" | bc`
243         printf "%0.4f " $dist_um >> dist_nuclear_envelope.txt
244     done
245     printf "\n" >> dist_nuclear_envelope.txt
246 done
247
248
249 #cleanup
250 #rm -rf nucleus_???.mod*

```

### C.3.20. sbem\_convexHull.sh

```

1     #!/bin/bash
2
3     ####
4     #
5     # sbem_convexHull.sh
6     #
7     # Arguments: -i STRING      Name of input model file to generate convex hull model file
for

```

```

8      #      -s STRING      Name of MRC stack that input model is derived from
9      #
10     # Example: sbem_convexHull -i SCN-
VL_WT_LD_ZT04_001_nucleus_003_2Dbin4_tc.subv -s ../../reconstruction/subv_nuclei/SCN-
VL_WT_LD_ZT04_001_nucleus_003_2Dbin4.mrc
11     #
12     ##=##=##=##=##=##=##=##=##
13
14     # Parse command line arguments
15     while getopts i:s: option
16     do
17         case "${option}"
18         in
19             i) INPUT=${OPTARG};;
20             s) MRC_stack=${OPTARG};;
21         esac
22     done
23
24     BASENAME=`echo ${INPUT}/.*/*`
25
26     mkdir temp_ch_mask
27     echo "Generating binary mask..."
28     imodmop -mask 1 ${INPUT} ${MRC_stack} ./temp_ch_mask/mask.mrc #Generate binary
mask
29     echo "Converting binary mask to a TIFF series..."
30     mrc2tif ./temp_ch_mask/mask.mrc ./temp_ch_mask/mask #Convert binary mask to a
series of TIF files
31
32     cd temp_ch_mask
33     N_tifs=`ls mask*.tif | wc -l`
34     echo "Computing the convex hull..."
35     cp /home/aperez/mfiles/sbem_convexHull.m .
36     grep -rl 'N_images=;' sbem_convexHull.m | xargs sed -i
's|N_images=;|N_images='${N_tifs}';|g'
37     matlab -nojvm -nosplash -nodesktop -nodisplay -r sbem_convexHull
38
39     echo "Converting to IMOD format..."
40     mv temp_ch_points.txt ..
41     cd ..
42     point2model -image ${MRC_stack} -color 255,0,0 temp_ch_points.txt
${BASENAME}_convexHull.subv
43     imodjoin ${INPUT} ${BASENAME}_convexHull.subv
${BASENAME}_convexHullJoin.subv
44     rm -rf temp_ch* *_convexHull.subv~

```

### C.3.21. sbem\_convexHull.m

```

1      N_images=;
2      N_images=N_images-1;
3
4      fid=fopen('temp_ch_points.txt','w');
5      C=1;
6      for N=0:N_images
7          N_str=sprintf('%0*d',3,N);

```



```

8     INPUT=['mask.' N_str '.tif'];
9     fprintf('Computing convex hull for Z=%s...\n',num2str(N));
10    I=imread(sprintf('%s',INPUT));
11    I=imrotate(I,-90);
12    OUTPUT=zeros(size(I));
13    ONES=find(I == 1);
14    if numel(ONES) > 3
15        [X,Y]=ind2sub(size(I),ONES);
16        DT=DelaunayTri(X,Y);
17        CH=convexHull(DT);
18        X_CH=DT.X(CH,1);
19        Y_CH=DT.X(CH,2);
20        Y_max=size(I,1);
21        Y_CH_imod=Y_CH;
22        P=[X_CH Y_CH_imod N*ones(size(X_CH))];
23        for i=1:size(P,1)
24            fprintf(fid,'%g %f %f %g\n',C,P(i,1),P(i,2),P(i,3));
25        end
26        C=C+1;
27    end
28 end
29 fclose(fid);
30 exit;

```

### C.3.22. totalCurvature.sh

```

1     #! /bin/bash
2
3     ###-##-##-##-##-##-##-##-##-##
4     #
5     # totalCurvature.sh
6     #
7     # Arguments: -i STRING    Name of subvolume being analyzed
8     #
9     # Example: sbem_totalCurvature -i SCN-
VL_WT_LD_ZT04_001_nucleus_008_2Dbin4.subv
10    #
11    ###-##-##-##-##-##-##-##-##-##
12
13    # Parse command line arguments
14    while getopts i: option
15    do
16        case "${option}"
17        in
18            i) INPUT_mod=${OPTARG};;
19        esac
20    done
21
22    if [ ! -d ../sbem_totalCurvature ]; then mkdir ../sbem_totalCurvature; fi
23
24    BASENAME=`echo ${INPUT_mod}/.*/*` #Extract basename
25
26    echo ${INPUT_mod}

```

```

27     echo "Converting to VRML format..."
28
29     cp ${INPUT_mod} ..
30     cd ..
31     cp sbem_convexHull/${BASENAME}_convexHull.subv .
32     imod2vrml2 -l ${INPUT_mod} ${BASENAME}.wml > /dev/null
33     imod2vrml2 -l ${BASENAME}_convexHull.subv ${BASENAME}_convexHull.wml >
/dev/null
34     rm -rf ${INPUT_mod} ${BASENAME}_convexHull.subv
35
36     # Create Amira network file
37     echo "Generating recomputed meshes and curvature vector/scalar fields and creating
movies..."
38     cp /home/aperez/amira_scripts/totalCurvature_withMovies.hx
./sbem_totalCurvature/${BASENAME}_curvature.hx
39     grep -rl 'BASENAME' ./sbem_totalCurvature/${BASENAME}_curvature.hx | xargs sed -i
's|BASENAME|${BASENAME}|g'
40     cd sbem_totalCurvature
41     mkdir tif_Curvedness tif_GaussCurvature tif_MaxCurvature tif_MeanCurvature
tif_ShapelIndex tif_MinCurvature tif_BendingEnergy tif_EulerCharacteristic
42
43     /ncmir/local.linux.amd64/Amira-5.4/bin/start ${BASENAME}_curvature.hx
44
45     N=`sed -n '4p' ${BASENAME}_remeshed.inp | cut -d ' ' -f1` #Extract number of vertices
(N)
46     M=`sed -n '4p' ${BASENAME}_remeshed.inp | cut -d ' ' -f2` #Extract number of faces (M)
47     N_sphere=`sed -n '4p' ${BASENAME}_sphere_remeshed.inp | cut -d ' ' -f1`
48     M_sphere=`sed -n '4p' ${BASENAME}_sphere_remeshed.inp | cut -d ' ' -f2`
49     echo "N vertices nucleus: ${N}"
50     echo "N faces nucleus:  ${M}"
51     echo "N vertices sphere: ${N_sphere}"
52     echo "N faces sphere:   ${M_sphere}"
53
54     # Create animated GIFs from output
55     for file in tif_*/*.tif; do convert $file ${file%.tif}.gif; done
56     gifsicle -l --colors 256 ./tif_Curvedness/*.gif > ${BASENAME}_Curvedness.gif
57     gifsicle -l --colors 256 ./tif_GaussCurvature/*.gif > ${BASENAME}_GaussCurvature.gif
58     gifsicle -l --colors 256 ./tif_MaxCurvature/*.gif > ${BASENAME}_MaxCurvature.gif
59     gifsicle -l --colors 256 ./tif_MeanCurvature/*.gif > ${BASENAME}_MeanCurvature.gif
60     gifsicle -l --colors 256 ./tif_ShapelIndex/*.gif > ${BASENAME}_ShapelIndex.gif
61     gifsicle -l --colors 256 ./tif_MinCurvature/*.gif > ${BASENAME}_MinCurvature.gif
62     gifsicle -l --colors 256 ./tif_BendingEnergy/*.gif > ${BASENAME}_BendingEnergy.gif
63     gifsicle -l --colors 256 ./tif_EulerCharacteristic/*.gif >
${BASENAME}_EulerCharacteristic.gif
64     rm -rf tif_Curvedness tif_GaussCurvature tif_MaxCurvature tif_MeanCurvature
tif_ShapelIndex tif_MinCurvature tif_BendingEnergy tif_EulerCharacteristic
65
66     echo "Formatting output..."
67     grep -v 'tri' ${BASENAME}_remeshed.inp >> temp_tc_verts.txt #Extract coordinates of
vertices by matching everything without tri
68     grep -i 'tri' ${BASENAME}_remeshed.inp >> temp_tc_tris.txt #Extract vertex identifiers for
each trinagle by matching everything with tri
69     grep -v 'tri' ${BASENAME}_sphere_remeshed.inp >> temp_tc_sphere_verts.txt
70     grep -i 'tri' ${BASENAME}_sphere_remeshed.inp >> temp_tc_sphere_tris.txt

```

```

71
72 N_lines=`cat temp_tc_verts.txt | wc -l`
73 N_lines_sphere=`cat temp_tc_sphere_verts.txt | wc -l`
74 sed -n '5,${N_lines}'p' temp_tc_verts.txt >> temp_tc_verts2.txt #Remove AVS UCD
format header information
75 sed -n '5,${N_lines_sphere}'p' temp_tc_sphere_verts.txt >> temp_tc_sphere_verts2.txt
76
77 # Remove extraneous information at the start, leaving just the three vertex identifiers
78 while read line
79 do
80     echo ""echo ${line} | cut -d ' ' -f4-6"" >> temp_tc_tris2.txt
81 done < temp_tc_tris.txt
82
83 while read line
84 do
85     echo ""echo ${line} | cut -d ' ' -f4-6"" >> temp_tc_sphere_tris2.txt
86 done < temp_tc_sphere_tris.txt
87
88 # Extract curvature values from Amira mesh ASCII files
89 sed -e '1,/d' ${BASENAME}_MaxCurvature.am >> temp_tc_max.txt
90 sed -e '1,/d' temp_tc_max.txt >> temp_tc_max2.txt
91 head -n -1 temp_tc_max2.txt >> ${BASENAME}_K_Max.txt #Remove final blank line
92
93 sed -e '1,/d' ${BASENAME}_MeanCurvature.am >> temp_tc_mean.txt
94 sed -e '1,/d' temp_tc_mean.txt >> temp_tc_mean2.txt
95 head -n -1 temp_tc_mean2.txt >> ${BASENAME}_K_Mean.txt
96
97 sed -e '1,/d' ${BASENAME}_GaussCurvature.am >> temp_tc_gauss.txt
98 sed -e '1,/d' temp_tc_gauss.txt >> temp_tc_gauss2.txt
99 head -n -1 temp_tc_gauss2.txt >> ${BASENAME}_K_Gaussian.txt
100
101 sed -e '1,/d' ${BASENAME}_ShapelIndex.am >> temp_tc_shape.txt
102 sed -e '1,/d' temp_tc_shape.txt >> temp_tc_shape2.txt
103 head -n -1 temp_tc_shape2.txt >> ${BASENAME}_s.txt
104
105 sed -e '1,/d' ${BASENAME}_Curvedness.am >> temp_tc_curve.txt
106 sed -e '1,/d' temp_tc_curve.txt >> temp_tc_curve2.txt
107 head -n -1 temp_tc_curve2.txt >> ${BASENAME}_C.txt
108
109 sed -e '1,/d' ${BASENAME}_BendingEnergy.am >> temp_tc_be.txt
110 sed -e '1,/d' temp_tc_be.txt >> temp_tc_be2.txt
111 head -n -1 temp_tc_be2.txt >> ${BASENAME}_EB.txt
112
113 sed -e '1,/d' ${BASENAME}_EulerCharacteristic.am >> temp_tc_x.txt
114 sed -e '1,/d' temp_tc_x.txt >> temp_tc_x2.txt
115 head -n -1 temp_tc_x2.txt >> ${BASENAME}_X.txt
116
117 sed -e '1,/d' ${BASENAME}_MinCurvature.am >> temp_tc_min.txt
118 sed -e '1,/d' temp_tc_min.txt >> temp_tc_min2.txt
119 head -n -1 temp_tc_min2.txt >> ${BASENAME}_K_Min.txt
120
121 sed -e '1,/d' ${BASENAME}_sphere_MaxCurvature.am >> temp_tc_sphere_max.txt
122 sed -e '1,/d' temp_tc_sphere_max.txt >> temp_tc_sphere_max2.txt
123 head -n -1 temp_tc_sphere_max2.txt >> ${BASENAME}_K_Max_sphere.txt

```

```

124
125 sed -e '1,/d' ${BASENAME}_sphere_MeanCurvature.am >>
temp_tc_sphere_mean.txt
126 sed -e '1,/d' temp_tc_sphere_mean.txt >> temp_tc_sphere_mean2.txt
127 head -n -1 temp_tc_sphere_mean2.txt >> ${BASENAME}_K_Mean_sphere.txt
128
129 sed -e '1,/d' ${BASENAME}_sphere_GaussCurvature.am >>
temp_tc_sphere_gauss.txt
130 sed -e '1,/d' temp_tc_sphere_gauss.txt >> temp_tc_sphere_gauss2.txt
131 head -n -1 temp_tc_sphere_gauss2.txt >> ${BASENAME}_K_Gaussian_sphere.txt
132
133 sed -e '1,/d' ${BASENAME}_sphere_ShapeIndex.am >> temp_tc_sphere_shape.txt
134 sed -e '1,/d' temp_tc_sphere_shape.txt >> temp_tc_sphere_shape2.txt
135 head -n -1 temp_tc_sphere_shape2.txt >> ${BASENAME}_s_sphere.txt
136
137 sed -e '1,/d' ${BASENAME}_sphere_Curvedness.am >> temp_tc_sphere_curve.txt
138 sed -e '1,/d' temp_tc_sphere_curve.txt >> temp_tc_sphere_curve2.txt
139 head -n -1 temp_tc_sphere_curve2.txt >> ${BASENAME}_C_sphere.txt
140
141 sed -e '1,/d' ${BASENAME}_sphere_BendingEnergy.am >> temp_tc_sphere_be.txt
142 sed -e '1,/d' temp_tc_sphere_be.txt >> temp_tc_sphere_be2.txt
143 head -n -1 temp_tc_sphere_be2.txt >> ${BASENAME}_EB_sphere.txt
144
145 sed -e '1,/d' ${BASENAME}_sphere_EulerCharacteristic.am >> temp_tc_sphere_x.txt
146 sed -e '1,/d' temp_tc_sphere_x.txt >> temp_tc_sphere_x2.txt
147 head -n -1 temp_tc_sphere_x2.txt >> ${BASENAME}_X_sphere.txt
148
149 sed -e '1,/d' ${BASENAME}_sphere_MinCurvature.am >> temp_tc_sphere_min.txt
150 sed -e '1,/d' temp_tc_sphere_min.txt >> temp_tc_sphere_min2.txt
151 head -n -1 temp_tc_sphere_min2.txt >> ${BASENAME}_K_Min_sphere.txt
152
153 mv temp_tc_tris2.txt ${BASENAME}_meshIndices.txt #Rename
154 mv temp_tc_verts2.txt ${BASENAME}_meshVertices.txt
155 mv temp_tc_sphere_tris2.txt ${BASENAME}_meshIndices_sphere.txt
156 mv temp_tc_sphere_verts2.txt ${BASENAME}_meshVertices_sphere.txt
157
158 rm -rf temp_tc_*.txt #Remove temporary files #Remove temporary files
159
160 mkdir ${BASENAME}_files #Create directory for files
161 sed -n '1,110p' ${BASENAME}_curvature.hx | grep -v 'save' >>
${BASENAME}_curvature_noSave.hx
162 mv ${BASENAME}* ${BASENAME}_files >> /dev/null 2>&1 #Move intermediate files to
_files directory
163 mkdir ${BASENAME}_movies
164 mv ${BASENAME}_files/*gif ${BASENAME}_files/*.mpg ${BASENAME}_movies
165 #mv ../${BASENAME}*.wml .
166
167
168 #Create MATLAB mfile to perform surface integrals
169 echo "Computing surface integrals..."
170 cp /home/aperez/mfiles/totalCurvature.m totalCurvature.m
171 grep -rl 'BASENAME' totalCurvature.m | xargs sed -i 's|BASENAME|${BASENAME}|g'
172

```

```

173 matlab -nojvm -nosplash -nodesktop -nodisplay -r totalCurvature #> /dev/null 2>&1 #Run
mfile
174 mv totalCurvature.m ./${BASENAME}_files
175 rm -rf *~
176 echo "Done."

```

### C.3.23. totalCurvature\_withMovies.hx

```

1      # Amira Script
2
3      viewer setBackgroundColor 0.5 0.5 0.5; #Make background grey
4      viewer setSize 900 900; #Set viewer size
5
6      #####
7      # Load the nucleus, remesh, and export
8      #####
9
10     load ../BASENAME.wml; #Load nucleus VRML file
11     create HxGeometryToSurface {lvToSurface}; #Create Inventor-to-surf format conversion
module
12     lvToSurface data connect BASENAME.wml; #Connect converter to VRML file
13     lvToSurface action setState index 0;
14     lvToSurface action touch 0;
15     lvToSurface fire; #Apply
16     GeometrySurface save "AVS UCD ascii" BASENAME.inp; #Save surface model in AVS
UCD format
17     create HxRemeshSurface; #Create RemeshSurface module
18     RemeshSurface data connect BASENAME.inp; #Connect to original surface
19     RemeshSurface desiredSize setValue 2 100; #Set percentage of triangles to keep at
100%
20     RemeshSurface remeshOptions1 setValue 0 1; #Turn on contour correction
21     RemeshSurface remeshOptions1 setValue 1 0;
22     RemeshSurface remesh setState index 0;
23     RemeshSurface remesh touch 0;
24     RemeshSurface fire; #Apply
25     BASENAME.remeshed save "AVS UCD ascii" BASENAME_remeshed.inp; #Save
remeshed model in AVS UCD format
26
27     #####
28     # Load the convex hull, remesh, and compute surface area
29     #####
30
31     load ../BASENAME_convexHull.wml;
32     create HxGeometryToSurface {lvToSurface};
33     lvToSurface2 data connect BASENAME_convexHull.wml;
34     lvToSurface2 action setState index 0;
35     lvToSurface2 action touch 0;
36     lvToSurface2 fire;
37     GeometrySurface save "AVS UCD ascii" BASENAME_convexHull.inp;
38     create HxRemeshSurface;
39     RemeshSurface2 data connect BASENAME_convexHull.inp;
40     RemeshSurface2 desiredSize setValue 2 100;
41     RemeshSurface2 remeshOptions1 setValue 0 1;
42     RemeshSurface2 remeshOptions1 setValue 1 0;

```

```

43   RemeshSurface2 remesh setState index 0;
44   RemeshSurface2 remesh touch 0;
45   RemeshSurface2 fire;
46   BASENAME_convexHull.remeshed save "AVS UCD ascii"
BASENAME_convexHull_remeshed.inp;
47   create HxSurfaceArea {Surface Area};
48   Surface-Area data connect BASENAME_convexHull_remeshed.inp;
49   Surface-Area action setState index 0;
50   Surface-Area action touch 0;
51   Surface-Area fire;
52   set A_CH [BASENAME_convexHull_remeshed.statistics getValue table1 2 1]; #Get
surface area of convex hull
53
54   #####
55   # Calculate and export curvature fields for the nucleus
56   #####
57
58   create HxGetCurvature; #Create GetCurvature module
59   GetCurvature data connect BASENAME_remeshed.inp; #Connect to remeshed surface
60   GetCurvature output setValue 6; #Compute both principal curvatures
61   GetCurvature method setValue 1;
62   GetCurvature create;
63   create HxArithmetic; #Maximum principal curvature
64   Arithmetic inputA connect BothCurvatures; #Connect Arithmetic module to curvatures
65   Arithmetic resultChannels setValue 1;
66   Arithmetic expr0 setValue Ar;
67   Arithmetic resultType setValue 0;
68   Arithmetic create;
69   Result save "Amiramesh ascii" BASENAME_MaxCurvature.am; #Save maximum
principal curvatures as Amira mesh ASCII file
70   create HxArithmetic; #Minimum principal curvature
71   Arithmetic2 inputA connect BothCurvatures;
72   Arithmetic2 resultChannels setValue 1;
73   Arithmetic2 expr0 setValue Ai;
74   Arithmetic2 resultType setValue 0;
75   Arithmetic2 create;
76   Result save "Amiramesh ascii" BASENAME_MinCurvature.am; #Save minimum principal
curvatures as Amira mesh ASCII file
77   create HxArithmetic; #Local bending energy
78   Arithmetic3 inputA connect BASENAME_MaxCurvature.am;
79   Arithmetic3 inputB connect BASENAME_MinCurvature.am;
80   Arithmetic3 resultChannels setValue 1;
81   Arithmetic3 resultType setValue 0;
82   Arithmetic3 expr0 setValue pow(A,2)+pow(B,2);
83   Arithmetic3 create;
84   Result save "Amiramesh ascii" BASENAME_BendingEnergy.am;
85   GetCurvature output setValue 2; #Set to Mean Curvature
86   GetCurvature method setValue 1;
87   GetCurvature create;
88   GetCurvature output setValue 4; #Set to Gaussian Curvature
89   GetCurvature method setValue 1;
90   GetCurvature create;
91   GetCurvature output setValue 10; #Set to ShapeIndex
92   GetCurvature method setValue 1;

```



```

142 RemeshSurface3 desiredSize setValue 2 100; #Set percentage of triangles to keep at
100%
143 RemeshSurface3 remeshOptions1 setValue 0 1;
144 RemeshSurface3 remeshOptions1 setValue 1 0;
145 RemeshSurface3 remesh setState index 0;
146 RemeshSurface3 remesh touch 0;
147 RemeshSurface3 fire; #Apply
148 BASENAME_sphere.remeshed save "AVS UCD ascii"
BASENAME_sphere_remeshed.inp; #Save remeshed sphere in AVS UCD format
149
150 #####
151 # Write volume/radius of sphere and convex hull area to files
152 #####
153
154 set data "$V $r"; #Set string to print
155 set filename "BASENAME_sphereRadius.txt"; #Set filename to save to
156 set fileld [open $filename "w"]; #Open file
157 puts $fileld $data; #Write data to file
158 close $fileld; #Close file
159 set data "$A_CH";
160 set filename "BASENAME_convexHullArea.txt";
161 set fileld [open $filename "w"];
162 puts $fileld $data;
163 close $fileld;
164
165 #####
166 # Calculate and export curvature fields for the sphere
167 #####
168
169 create HxGetCurvature; #Create GetCurvature module
170 GetCurvature2 data connect BASENAME_sphere_remeshed.inp; #Connect to remeshed
surface
171 GetCurvature2 output setValue 6; #Compute both principal curvatures
172 GetCurvature2 method setValue 1;
173 GetCurvature create;
174 create HxArithmetic; #Maximum principal curvature
175 Arithmetic5 inputA connect BothCurvatures; #Connect Arithmetic module to curvatures
176 Arithmetic5 resultChannels setValue 1;
177 Arithmetic5 expr0 setValue Ar;
178 Arithmetic5 resultType setValue 0;
179 Arithmetic5 create;
180 Result save "Amiramesh ascii" BASENAME_sphere_MaxCurvature.am; #Save maximum
principal curvatures as Amira mesh ASCII file
181 create HxArithmetic; #Minimum principal curvature
182 Arithmetic6 inputA connect BothCurvatures;
183 Arithmetic6 resultChannels setValue 1;
184 Arithmetic6 expr0 setValue Ai;
185 Arithmetic6 resultType setValue 0;
186 Arithmetic6 create;
187 Result save "Amiramesh ascii" BASENAME_sphere_MinCurvature.am; #Save minimum
principal curvatures as Amira mesh ASCII file
188 create HxArithmetic; #Local bending energy
189 Arithmetic7 inputA connect BASENAME_MaxCurvature.am;
190 Arithmetic7 inputB connect BASENAME_MinCurvature.am;

```



```

191 Arithmetic7 resultChannels setValue 1;
192 Arithmetic7 resultType setValue 0;
193 Arithmetic7 expr0 setValue pow(A,2)+pow(B,2);
194 Arithmetic7 create;
195 Result save "Amiramesh ascii" BASENAME_sphere_BendingEnergy.am;
196 GetCurvature2 output setValue 2;
197 GetCurvature2 method setValue 1;
198 GetCurvature2 create;
199 GetCurvature2 output setValue 4;
200 GetCurvature2 method setValue 1;
201 GetCurvature2 create;
202 GetCurvature2 output setValue 10;
203 GetCurvature2 method setValue 1;
204 GetCurvature2 dolt setState index 0;
205 GetCurvature2 dolt touch 0;
206 GetCurvature2 fire;
207 GetCurvature2 output setValue 11;
208 GetCurvature2 method setValue 1;
209 GetCurvature2 dolt setState index 0;
210 GetCurvature2 dolt touch 0;
211 GetCurvature2 fire;
212 GaussCurvature save "Amiramesh ascii" BASENAME_sphere_GaussCurvature.am;
213 MeanCurvature save "Amiramesh ascii" BASENAME_sphere_MeanCurvature.am;
214 ShapelIndex save "Amiramesh ascii" BASENAME_sphere_ShapelIndex.am;
215 Curvedness save "Amiramesh ascii" BASENAME_sphere_Curvedness.am;
216 create HxArithmetic; #Euler characteristic
217 Arithmetic8 inputA connect BASENAME_GaussCurvature.am;
218 Arithmetic8 resultChannels setValue 1;
219 Arithmetic8 resultType setValue 0;
220 Arithmetic8 expr0 setValue A/(2*2*cos(0));
221 Arithmetic8 create;
222 Result save "Amiramesh ascii" BASENAME_sphere_EulerCharacteristic.am;
223
224 #####
225 # Create movies for Max Curvature
226 #####
227
228 SurfaceView setViewerMask 0; #Turn off previous SurfaceViews
229 SurfaceView2 setViewerMask 0;
230 create HxDisplaySurface {SurfaceView3}; #Create new SurfaceView module
231 SurfaceView3 data connect BASENAME_remeshed.inp; #Connect to nucleus surface
232 SurfaceView3 colorField connect BASENAME_MaxCurvature.am; #Connect to
MaxCurvature scalar field
233 SurfaceView3 drawStyle setValue 0; #Set draw style to outline triangles
234 SurfaceView3 drawStyle setOutlineColor 0 0 0; #Set outline color to black
235 SurfaceView3 drawStyle setState 0 1 1 1 0 1; #Set draw style to vertex normals
236 [load ${AMIRA_ROOT}/data/colormaps/physics.icol] setLabel physics2.icol; #Load
physics colormap
237 physics2.icol Datafield connect BASENAME_MaxCurvature.am; #Connect colormap to
MaxCurvature scalar field
238 physics2.icol setAlphaCurve 0; #Set alpha to zero
239 physics2.icol select; #Apply
240 SurfaceView3 colormap connect physics2.icol; #Connect SurfaceView to physics
colormap

```

```

241   SurfaceView3 colormap setLocalRange 0; #Set colormap range to that of the
MaxCurvature scalar field
242   SurfaceView3 buffer touch 4; #Apply changes
243   SurfaceView3 fire;
244   create HxDisplayColormap; #Create color map display
245   DisplayColormap data connect physics2.icol; #Connect to physics colormap
246   DisplayColormap options setValue 1 1; #Make display vertical
247   DisplayColormap options setValue 3 1; #Make background transparent
248   DisplayColormap setColor black; #Make text black
249   DisplayColormap size setValue 300; #Make colormap display longer
250   DisplayColormap select; #Apply
251   create HxAnnotation; #Create annotation
252   set INPUT [BASENAME.wml getLabel]; #Read in name of input wml file to variable
INPUT
253   Annotation text setValue "Maximum Principal Curvature"; #Set text of annotation
254   Annotation font setFontName Helvetica;
255   Annotation font setFontSize 24;
256   Annotation font setFontColor black;
257   Annotation fire; #Apply
258   create HxAnnotation;
259   Annotation2 text setValue "$INPUT";
260   Annotation2 font setFontName Helvetica;
261   Annotation2 font setFontSize 14;
262   Annotation2 font setFontColor black;
263   Annotation2 position setValue 0 450;
264   Annotation2 position setValue 1 5;
265   Annotation2 fire;
266
267   create HxCircularCameraPath {CameraRotate}; #Create CameraRotate module
268   create HxScriptObject {DemoMaker}; #Create DemoMaker module
269   create HxMovieMaker {MovieMaker}; #Create MovieMaker module
270   DemoMaker script setValue ${AMIRA_ROOT}/share/script-
objects/DemoMakerClassic.scro; #Load classic DemoMaker
271   DemoMaker setVar scroTypeDemoMaker 1; #Set variables required for DemoMaker to
work
272   DemoMaker setVar isInitialized 1;
273   DemoMaker setVar isDemoMakerActive 1;
274   DemoMaker setVar funcKeysDefined 1;
275   DemoMaker setVar lastStartTime 4; #Start time of last event
276   DemoMaker setVar lastEndTime 8; #End time of last event (i.e., max time)
277   DemoMaker setVar lastTimeStep 0;
278   DemoMaker setVar internalEventList {dummy {numeric CameraRotate/Time 0 4 0 360 0
360 {CameraRotate time setValue %0%; CameraRotate fire}} {select CameraRotate/Action 4 1 0
{{most vertical} x-axis y-axis z-axis {up direction}} {CameraRotate action setOptValue %0%;
CameraRotate fire}} {button CameraRotate/Action/recompute 4 0 0 0 {
279       if %0% {CameraRotate action setShiftDown}
280       if %1% {CameraRotate action setCtrlDown}
281       if %2% {CameraRotate action setAltDown}
282       CameraRotate action setValue 0
283       CameraRotate fire
284       }} {numeric CameraRotate/Time 4 8 0 360 0 360 {CameraRotate time setValue
%0%; CameraRotate fire}}}; #Copy and paste from output
285   DemoMaker fire; #Apply
286   DemoMaker time setMinMax 0 8; #Set min and max times (min will always be 0)

```

```

287
288 MovieMaker time connect DemoMaker; #Connect to DemoMaker
289 MovieMaker fileFormat setValue 0; #Set output as MPEG movie
290 MovieMaker frames setValue 240; #Set number of frames (such that time is equal to max
time)
291 MovieMaker compressionQuality setValue 1.0; #Set compression quality to 1 (max
quality)
292 MovieMaker filename setValue BASENAME_MaxCurvature.mpg; #Set name of output
293 MovieMaker action setState index 0;
294 MovieMaker action touch 0;
295 MovieMaker fire; #Start recording
296 MovieMaker fileFormat setValue 2; #Set output as TIF series to create animated GIF
from
297 MovieMaker frames setValue 50; #Lower number of frames
298 MovieMaker filename setValue tif_MaxCurvature/BASENAME_MaxCurvature.tif; #Set
filename for output series
299 MovieMaker action setState index 0;
300 MovieMaker action touch 0;
301 MovieMaker fire; #Start recording
302
303 #####
304 ## Create movies for Min Curvature
305 #####
306
307 physics2.icol deselect;
308 DisplayColormap deselect;
309 SurfaceView3 colorField connect BASENAME_MinCurvature.am;
310 physics2.icol Datafield connect BASENAME_MinCurvature.am;
311 physics2.icol select;
312 SurfaceView3 colormap connect physics2.icol;
313 SurfaceView3 colormap setLocalRange 0;
314 SurfaceView3 buffer touch 4;
315 SurfaceView3 fire;
316 DisplayColormap data connect physics2.icol;
317 DisplayColormap select;
318 Annotation text setValue "Minimum Principial Curvature";
319 Annotation fire;
320
321 MovieMaker fileFormat setValue 0;
322 MovieMaker frames setValue 240;
323 MovieMaker compressionQuality setValue 1.0;
324 MovieMaker filename setValue BASENAME_MinCurvature.mpg;
325 MovieMaker action setState index 0;
326 MovieMaker action touch 0;
327 MovieMaker fire;
328 MovieMaker fileFormat setValue 2;
329 MovieMaker frames setValue 50;
330 MovieMaker filename setValue tif_MinCurvature/BASENAME_MinCurvature.tif;
331 MovieMaker action setState index 0;
332 MovieMaker action touch 0;
333 MovieMaker fire;
334
335 #####
336 # Create movies for Curvedness

```

```
337 #####
338
339 physics2.icol deselect;
340 DisplayColormap deselect;
341 SurfaceView3 colorField connect BASENAME_Curvedness.am;
342 physics2.icol Datafield connect BASENAME_Curvedness.am;
343 physics2.icol select;
344 SurfaceView3 colormap connect physics2.icol;
345 SurfaceView3 colormap setLocalRange 0;
346 SurfaceView3 buffer touch 4;
347 SurfaceView3 fire;
348 DisplayColormap data connect physics2.icol;
349 DisplayColormap select;
350 Annotation text setValue "Local Curvedness";
351 Annotation fire;
352
353 MovieMaker fileFormat setValue 0;
354 MovieMaker frames setValue 240;
355 MovieMaker compressionQuality setValue 1.0;
356 MovieMaker filename setValue BASENAME_Curvedness.mpg;
357 MovieMaker action setState index 0;
358 MovieMaker action touch 0;
359 MovieMaker fire;
360 MovieMaker fileFormat setValue 2;
361 MovieMaker frames setValue 50;
362 MovieMaker filename setValue tif_Curvedness/BASENAME_Curvedness.tif;
363 MovieMaker action setState index 0;
364 MovieMaker action touch 0;
365 MovieMaker fire;
366
367 #####
368 # Create movies for Mean Curvature
369 #####
370
371 physics2.icol deselect;
372 DisplayColormap deselect;
373 SurfaceView3 colorField connect BASENAME_MeanCurvature.am;
374 physics2.icol Datafield connect BASENAME_MeanCurvature.am;
375 physics2.icol select;
376 SurfaceView3 colormap connect physics2.icol;
377 SurfaceView3 colormap setLocalRange 0;
378 SurfaceView3 buffer touch 4;
379 SurfaceView3 fire;
380 DisplayColormap data connect physics2.icol;
381 DisplayColormap select;
382 Annotation text setValue "Local Mean Curvature";
383 Annotation fire;
384
385 MovieMaker fileFormat setValue 0;
386 MovieMaker frames setValue 240;
387 MovieMaker compressionQuality setValue 1.0;
388 MovieMaker filename setValue BASENAME_MeanCurvature.mpg;
389 MovieMaker action setState index 0;
390 MovieMaker action touch 0;
```

```

391 MovieMaker fire;
392 MovieMaker fileFormat setValue 2;
393 MovieMaker frames setValue 50;
394 MovieMaker filename setValue tif_MeanCurvature/BASENAME_MeanCurvature.tif;
395 MovieMaker action setState index 0;
396 MovieMaker action touch 0;
397 MovieMaker fire;
398
399 #####
400 # Create movies for Gaussian Curvature
401 #####
402
403 physics2.icol deselect;
404 DisplayColormap deselect;
405 SurfaceView3 colorField connect BASENAME_GaussCurvature.am;
406 physics2.icol Datafield connect BASENAME_GaussCurvature.am;
407 physics2.icol select;
408 SurfaceView3 colormap connect physics2.icol;
409 SurfaceView3 colormap setLocalRange 0;
410 SurfaceView3 buffer touch 4;
411 SurfaceView3 fire;
412 DisplayColormap data connect physics2.icol;
413 DisplayColormap select;
414 Annotation text setValue "Local Gaussian Curvature";
415 Annotation fire;
416
417 MovieMaker fileFormat setValue 0;
418 MovieMaker frames setValue 240;
419 MovieMaker compressionQuality setValue 1.0;
420 MovieMaker filename setValue BASENAME_GaussCurvature.mpg;
421 MovieMaker action setState index 0;
422 MovieMaker action touch 0;
423 MovieMaker fire;
424 MovieMaker fileFormat setValue 2;
425 MovieMaker frames setValue 50;
426 MovieMaker filename setValue tif_GaussCurvature/BASENAME_GaussCurvature.tif;
427 MovieMaker action setState index 0;
428 MovieMaker action touch 0;
429 MovieMaker fire;
430
431 #####
432 # Create movies for Shape Index
433 #####
434
435 physics2.icol deselect;
436 DisplayColormap deselect;
437 SurfaceView3 colorField connect BASENAME_ShapeIndex.am;
438 physics2.icol Datafield connect BASENAME_ShapeIndex.am;
439 physics2.icol select;
440 SurfaceView3 colormap connect physics2.icol;
441 SurfaceView3 colormap setLocalRange 0;
442 SurfaceView3 buffer touch 4;
443 SurfaceView3 fire;
444 DisplayColormap data connect physics2.icol;

```

```

445 DisplayColormap select;
446 Annotation text setValue "Local Shape Index";
447 Annotation fire;
448
449 MovieMaker fileFormat setValue 0;
450 MovieMaker frames setValue 240;
451 MovieMaker compressionQuality setValue 1.0;
452 MovieMaker filename setValue BASENAME_ShapelIndex.mpg;
453 MovieMaker action setState index 0;
454 MovieMaker action touch 0;
455 MovieMaker fire;
456 MovieMaker fileFormat setValue 2;
457 MovieMaker frames setValue 50;
458 MovieMaker filename setValue tif_ShapelIndex/BASENAME_ShapelIndex.tif;
459 MovieMaker action setState index 0;
460 MovieMaker action touch 0;
461 MovieMaker fire;
462
463 #####
464 ## Create movies for Bending Energy
465 #####
466
467 physics2.icol deselect;
468 DisplayColormap deselect;
469 SurfaceView3 colorField connect BASENAME_BendingEnergy.am;
470 physics2.icol Datafield connect BASENAME_BendingEnergy.am;
471 physics2.icol select;
472 SurfaceView3 colormap connect physics2.icol;
473 SurfaceView3 colormap setLocalRange 0;
474 SurfaceView3 buffer touch 4;
475 SurfaceView3 fire;
476 DisplayColormap data connect physics2.icol;
477 DisplayColormap select;
478 Annotation text setValue "Local Bending Energy";
479 Annotation fire;
480
481 MovieMaker fileFormat setValue 0;
482 MovieMaker frames setValue 240;
483 MovieMaker compressionQuality setValue 1.0;
484 MovieMaker filename setValue BASENAME_BendingEnergy.mpg;
485 MovieMaker action setState index 0;
486 MovieMaker action touch 0;
487 MovieMaker fire;
488 MovieMaker fileFormat setValue 2;
489 MovieMaker frames setValue 50;
490 MovieMaker filename setValue tif_BendingEnergy/BASENAME_BendingEnergy.tif;
491 MovieMaker action setState index 0;
492 MovieMaker action touch 0;
493 MovieMaker fire;
494
495 #####
496 ## Create movies for Euler Characteristic
497 #####
498

```

```

499 physics2.icol deselect;
500 DisplayColormap deselect;
501 SurfaceView3 colorField connect BASENAME_EulerCharacteristic.am;
502 physics2.icol Datafield connect BASENAME_EulerCharacteristic.am;
503 physics2.icol select;
504 SurfaceView3 colormap connect physics2.icol;
505 SurfaceView3 colormap setLocalRange 0;
506 SurfaceView3 buffer touch 4;
507 SurfaceView3 fire;
508 DisplayColormap data connect physics2.icol;
509 DisplayColormap select;
510 Annotation text setValue "Local Euler Characteristic";
511 Annotation fire;
512
513 MovieMaker fileFormat setValue 0;
514 MovieMaker frames setValue 240;
515 MovieMaker compressionQuality setValue 1.0;
516 MovieMaker filename setValue BASENAME_EulerCharacteristic.mpg;
517 MovieMaker action setState index 0;
518 MovieMaker action touch 0;
519 MovieMaker fire;
520 MovieMaker fileFormat setValue 2;
521 MovieMaker frames setValue 50;
522 MovieMaker filename setValue
tif_EulerCharacteristic/BASENAME_EulerCharacteristic.tif;
523 MovieMaker action setState index 0;
524 MovieMaker action touch 0;
525 MovieMaker fire;
526
527 exit;
528

```

### C.3.24. totalCurvature.m

```

1 % Load nucleus data
2 INDS=load('./BASENAME_files/BASENAME_meshIndices.txt');
3 VERTS=load('./BASENAME_files/BASENAME_meshVertices.txt');
4 K_max=load('./BASENAME_files/BASENAME_K_Max.txt');
5 K_mean=load('./BASENAME_files/BASENAME_K_Mean.txt');
6 K_gaussian=load('./BASENAME_files/BASENAME_K_Gaussian.txt');
7 s=load('./BASENAME_files/BASENAME_s.txt');
8 C=load('./BASENAME_files/BASENAME_C.txt');
9 X=load('./BASENAME_files/BASENAME_X.txt');
10 EB=load('./BASENAME_files/BASENAME_EB.txt');
11 K_min=load('./BASENAME_files/BASENAME_K_Min.txt');
12
13 % Load sphere data
14 INDS_sphere=load('./BASENAME_files/BASENAME_meshIndices_sphere.txt');
15 VERTS_sphere=load('./BASENAME_files/BASENAME_meshVertices_sphere.txt');
16 K_max_sphere=load('./BASENAME_files/BASENAME_K_Max_sphere.txt');
17 K_mean_sphere=load('./BASENAME_files/BASENAME_K_Mean_sphere.txt');
18 K_gaussian_sphere=load('./BASENAME_files/BASENAME_K_Gaussian_sphere.txt');
19 s_sphere=load('./BASENAME_files/BASENAME_s_sphere.txt');
20 C_sphere=load('./BASENAME_files/BASENAME_C_sphere.txt');

```

```

21 X_sphere=load('./BASENAME_files/BASENAME_X_sphere.txt');
22 EB_sphere=load('./BASENAME_files/BASENAME_EB_sphere.txt');
23 K_min_sphere=load('./BASENAME_files/BASENAME_K_Min_sphere.txt');
24
25 % Load convex hull area
26 A_CH=load('./BASENAME_files/BASENAME_convexHullArea.txt');
27
28 VERTS=VERTS(:,2:4); %Extract vertex coordinates only
29 INDS=INDS+1; %Change indices to MATLAB format (starting at 1 instead of 0)
30 VERTS_sphere=VERTS_sphere(:,2:4);
31 INDS_sphere=INDS_sphere+1;
32
33 %Compute surface integrals for nucleus
34 SUM_area=0;
35 SUM_max=0; SUM_mean=0; SUM_gauss=0; SUM_s=0; SUM_C=0; SUM_X=0;
SUM_EB=0; SUM_min=0;
36 for i=1:size(INDS,1)
37     P_A=VERTS(INDS(i,1),:);
38     P_B=VERTS(INDS(i,2),:);
39     P_C=VERTS(INDS(i,3),:);
40     AREA=0.5*norm(cross(P_C-P_A,P_C-P_B));
41     K_max_i=mean(K_max(INDS(i,:)));
42     K_mean_i=mean(K_mean(INDS(i,:)));
43     K_gaussian_i=mean(K_gaussian(INDS(i,:)));
44     s_i=mean(s(INDS(i,:)));
45     C_i=mean(C(INDS(i,:)));
46     X_i=mean(X(INDS(i,:)));
47     K_min_i=mean(K_min(INDS(i,:)));
48     EB_i=mean(EB(INDS(i,:)));
49     SUM_max=SUM_max+K_max_i*AREA;
50     SUM_mean=SUM_mean+K_mean_i*AREA;
51     SUM_gauss=SUM_gauss+K_gaussian_i*AREA;
52     SUM_s=SUM_s+s_i*AREA;
53     SUM_C=SUM_C+C_i*AREA;
54     SUM_X=SUM_X+X_i*AREA;
55     SUM_EB=SUM_EB+EB_i*AREA;
56     SUM_min=SUM_min+K_min_i*AREA;
57     SUM_area=SUM_area+AREA;
58 end
59
60 %Compute surface integrals for sphere
61 SUM_area_sphere=0;
62 SUM_max_sphere=0; SUM_mean_sphere=0; SUM_gauss_sphere=0;
SUM_s_sphere=0; SUM_C_sphere=0; SUM_X_sphere=0; SUM_EB_sphere=0;
SUM_min_sphere=0;
63 for i=1:size(INDS_sphere,1)
64     P_A=VERTS_sphere(INDS_sphere(i,1),:);
65     P_B=VERTS_sphere(INDS_sphere(i,2),:);
66     P_C=VERTS_sphere(INDS_sphere(i,3),:);
67     AREA=0.5*norm(cross(P_C-P_A,P_C-P_B));
68     K_max_sphere_i=mean(K_max_sphere(INDS_sphere(i,:)));
69     K_mean_sphere_i=mean(K_mean_sphere(INDS_sphere(i,:)));
70     K_gaussian_sphere_i=mean(K_gaussian_sphere(INDS_sphere(i,:)));
71     s_sphere_i=mean(s_sphere(INDS_sphere(i,:)));

```



```

72     C_sphere_i=mean(C_sphere(INDS_sphere(i,:)));
73     X_sphere_i=mean(X_sphere(INDS_sphere(i,:)));
74     EB_sphere_i=mean(EB_sphere(INDS_sphere(i,:)));
75     K_min_sphere_i=mean(K_min_sphere(INDS_sphere(i,:)));
76     SUM_max_sphere=SUM_max_sphere+K_max_sphere_i*AREA;
77     SUM_mean_sphere=SUM_mean_sphere+K_mean_sphere_i*AREA;
78     SUM_gauss_sphere=SUM_gauss_sphere+K_gaussian_sphere_i*AREA;
79     SUM_s_sphere=SUM_s_sphere+s_sphere_i*AREA;
80     SUM_C_sphere=SUM_C_sphere+C_sphere_i*AREA;
81     SUM_X_sphere=SUM_X_sphere+X_sphere_i*AREA;
82     SUM_EB_sphere=SUM_EB_sphere+EB_sphere_i*AREA;
83     SUM_min_sphere=SUM_min_sphere+K_min_sphere_i*AREA;
84     SUM_area_sphere=SUM_area_sphere+AREA;
85 end
86
87 %Compute normalized, average, absolute values for nucleus
88 Q=A_CH/SUM_area;
89 KnaaMax=Q*sum(abs(K_max));
90 KnaaMin=Q*sum(abs(K_min));
91 KnaaMean=Q*sum(abs(K_mean));
92 KnaaGauss=Q*sum(abs(K_gaussian));
93 Snaa=Q*sum(abs(s));
94 Cnaa=Q*sum(abs(C));
95 Xnaa=Q*sum(abs(X));
96 EBnaa=Q*sum(abs(EB));
97
98 %Compute normalized, average values for nucleus
99 KnaMax=Q*sum(K_max);
100 KnaMin=Q*sum(K_min);
101 KnaMean=Q*sum(K_mean);
102 KnaGauss=Q*sum(K_gaussian);
103 Sna=Q*sum(s);
104 Cna=Q*sum(C);
105 Xna=Q*sum(X);
106 EBna=Q*sum(EB);
107
108 fid=fopen('BASENAME_results.txt','w');
109 fprintf(fid,'K_max   %f %f %f %f %f
%f\n',SUM_max,SUM_max_sphere,SUM_max/SUM_max_sphere,SUM_max-
SUM_max_sphere,KnaaMax,KnaMax);
110 fprintf(fid,'K_min   %f %f %f %f %f
%f\n',SUM_min,SUM_min_sphere,SUM_min/SUM_min_sphere,SUM_min-
SUM_min_sphere,KnaaMin,KnaMin);
111 fprintf(fid,'K_mean  %f %f %f %f %f
%f\n',SUM_mean,SUM_mean_sphere,SUM_mean/SUM_mean_sphere,SUM_mean-
SUM_mean_sphere,KnaaMean,KnaMean);
112 fprintf(fid,'K_gauss %f %f %f %f %f
%f\n',SUM_gauss,SUM_gauss_sphere,SUM_gauss/SUM_gauss_sphere,SUM_gauss-
SUM_gauss_sphere,KnaaGauss,KnaGauss);
113 fprintf(fid,'s      %f %f %f %f %f
%f\n',SUM_s,SUM_s_sphere,SUM_s/SUM_s_sphere,SUM_s-SUM_s_sphere,Snaa,Sna);
114 fprintf(fid,'C      %f %f %f %f %f
%f\n',SUM_C,SUM_C_sphere,SUM_C/SUM_C_sphere,SUM_C-SUM_C_sphere,Cnaa,Cna);

```

```
115     fprintf(fid,'X      %f %f %f %f %f %f
%f\n',SUM_X,SUM_X_sphere,SUM_X/SUM_X_sphere,SUM_X-SUM_X_sphere,Xnaa,Xna);
116     fprintf(fid,'EB      %f %f %f %f %f %f
%f\n',SUM_EB,SUM_EB_sphere,SUM_EB/SUM_EB_sphere,SUM_EB-
SUM_EB_sphere,EBnaa,EBna);
117     fprintf(fid,'Area   %f\n',SUM_area);
118     fprintf(fid,'CH Area %f\n',A_CH);
119     fclose(fid);
120     exit;
```

## Appendix D. Example output from automated nuclear analysis.

Output of automatic nuclear analysis for the ZT04\_01 SCN SBEM dataset (CCDBID: 81739) – Contents of *nucleus\_morphology.txt*

Cell #	Nuclear Volume ( $\mu\text{m}^3$ )	Nuclear Surface Area ( $\mu\text{m}^2$ )	Nuclear Surface Area to Volume Ratio
1	328.289	270.48	0.8239
2	334.793	290.63	0.868
3	355.311	338.51	0.9527
4	353.863	330.25	0.9332
5	323.646	353.98	1.0937
6	298.639	274.52	0.9192
7	369.18	337.52	0.9142
8	361.215	295.76	0.8187
9	374.046	349.27	0.9337
10	328.484	276.18	0.8407
11	308.978	324.97	1.0517
12	354.269	336	0.9484
13	311.194	268.38	0.8624
14	332.515	269.09	0.8092
15	322.815	341.06	1.0565
16	299.195	284.22	0.9499
17	314.6	331.33	1.0531
18	400.399	309.5	0.7729
19	358.404	370.76	1.0344
20	303.43	295.95	0.9753
21	398.756	294.07	0.7374
22	265.793	328.58	1.2362
23	298.679	299.7	1.0034
24	325.7	303.42	0.9315
25	345.7	340.45	0.9848
26	276.831	240.17	0.8675
27	352.463	411.85	1.1684
28	381.835	322.97	0.8458
29	368.115	383.41	1.0415
30	289.182	296.71	1.026
31	292.87	333.76	1.1396
32	360.017	340.34	0.9453
33	342.072	356.4	1.0418
34	344.478	339.36	0.9851
35	393.73	390.18	0.9909
36	349.85	375.99	1.0747

Output of automatic nuclear analysis for the ZT04\_01 SCN SBEM dataset (CCDBID: 81739) – Contents of *nucleus\_morphology.txt*, Continued.

Cell #	Nuclear Volume ( $\mu\text{m}^3$ )	Nuclear Surface Area ( $\mu\text{m}^2$ )	Nuclear Surface Area to Volume Ratio
37	315.065	348.02	1.1045
38	350.649	332.14	0.9472
39	299.036	288.31	0.9641
40	283.925	307.25	1.0821
41	393.831	331.85	0.8426
42	320.992	298.6	0.9302
43	334.407	347.49	1.0391
44	353.524	301.5	0.8528
45	363.566	358.86	0.987
46	327.559	279.23	0.8524
47	378.257	339.51	0.8975
48	324.358	316.64	0.9762
49	324.3	315.48	0.9728
50	287.457	320.73	1.1157
51	349.838	304.34	0.8699
52	301.559	246.81	0.8184
53	290.557	262.98	0.905
54	262.882	239.85	0.9124
55	301.02	256.53	0.8522
56	370.761	378.57	1.021
57	358.79	308.62	0.8601
58	343.786	304.06	0.8844
59	352.9	278.81	0.79
60	361.938	329.45	0.9102
61	301.145	280.45	0.9312
62	334.374	368.31	1.1014
63	339.283	332.91	0.9812
64	330.531	336.58	1.0183
65	363.945	347.46	0.9546
66	288.755	303.51	1.051
67	339.294	289.15	0.8522
68	335.41	406.25	1.2112
69	342.512	281.84	0.8228
70	292.881	239.75	0.8185
71	304.81	317.88	1.0428
72	269.82	315.98	1.171
73	348.26	370.05	1.0625
74	316.729	303.97	0.9597

Output of automatic nuclear analysis for the ZT04\_01 SCN SBEM dataset (CCDBID: 81739) – Contents of *nucleus\_morphology.txt*, Continued.

Cell #	Nuclear Volume ( $\mu\text{m}^3$ )	Nuclear Surface Area ( $\mu\text{m}^2$ )	Nuclear Surface Area to Volume Ratio
75	372.935	340.63	0.9133
76	306.545	315.54	1.0293
77	315.727	248.48	0.787
78	288.008	280.11	0.9725
79	373.179	314.8	0.8435
80	330.767	305.78	0.9244
81	287.432	305.8	1.0639

Output of automatic nuclear analysis for the ZT04\_01 SCN SBEM dataset – Contents of *nucleolus\_morphology.txt*.

Cell #	# of Nucleoli	Volume: Nucleolus #1 ( $\mu\text{m}^3$ )	Volume: Nucleolus #2 ( $\mu\text{m}^3$ )	Volume: Nucleolus #3 ( $\mu\text{m}^3$ )	Volume: Nucleolus #4 ( $\mu\text{m}^3$ )	Total Nucleolar Volume ( $\mu\text{m}^3$ )	Nucleolar Volume Fraction
1	2	1.7118	0.8404	0	0	2.5522	0.0077
2	2	0.5372	2.8246	0	0	3.3618	0.01
3	2	2.9809	0.674	0	0	3.6549	0.0102
4	1	3.0072	0	0	0	3.0072	0.0084
5	1	3.6435	0	0	0	3.6435	0.0112
6	1	2.8928	0	0	0	2.8928	0.0096
7	2	2.47	0.7057	0	0	3.1757	0.0086
8	2	2.5347	0.5222	0	0	3.0569	0.0084
9	1	3.3999	0	0	0	3.3999	0.009
10	3	1.5026	1.0823	0.3535	0	2.9384	0.0089
11	1	3.138	0	0	0	3.138	0.0101
12	1	3.7283	0	0	0	3.7283	0.0105
13	1	3.2668	0	0	0	3.2668	0.0104
14	2	1.2899	1.1607	0	0	2.4506	0.0073
15	1	2.6785	0	0	0	2.6785	0.0082
16	3	0.814	0.5936	0.8032	0	2.2108	0.0073
17	1	2.7775	0	0	0	2.7775	0.0088
18	2	3.2294	1.08	0	0	4.3094	0.0107
19	2	0.5631	2.891	0	0	3.4541	0.0096
20	1	3.2697	0	0	0	3.2697	0.0107
21	3	0.7816	1.2201	1.0732	0	3.0749	0.0077
22	1	2.2766	0	0	0	2.2766	0.0085
23	1	2.7743	0	0	0	2.7743	0.0092
24	1	3.4016	0	0	0	3.4016	0.0104

Output of automatic nuclear analysis for the ZT04\_01 SCN SBEM dataset – Contents of *nucleolus\_morphology.txt*, Continued.

Cell #	# of Nucleoli	Volume: Nucleolus #1 ( $\mu\text{m}^3$ )	Volume: Nucleolus #2 ( $\mu\text{m}^3$ )	Volume: Nucleolus #3 ( $\mu\text{m}^3$ )	Volume: Nucleolus #4 ( $\mu\text{m}^3$ )	Total Nucleolar Volume ( $\mu\text{m}^3$ )	Nucleolar Volume Fraction
25	1	3.4951	0	0	0	3.4951	0.0101
26	1	2.7677	0	0	0	2.7677	0.0099
27	1	4.2747	0	0	0	4.2747	0.0121
28	3	0.9736	2.5109	0.8481	0	4.3326	0.0113
29	1	3.7762	0	0	0	3.7762	0.0102
30	2	1.527	0.8984	0	0	2.4254	0.0083
31	2	2.3923	0.8598	0	0	3.2521	0.0111
32	2	1.7329	1.7291	0	0	3.462	0.0096
33	3	1.857	0.3322	0.5357	0	2.7249	0.0079
34	2	1.0932	1.4355	0	0	2.5287	0.0073
35	1	4.2308	0	0	0	4.2308	0.0107
36	1	3.4081	0	0	0	3.4081	0.0097
37	2	2.3611	0.1844	0	0	2.5455	0.008
38	1	3.1578	0	0	0	3.1578	0.009
39	1	3.5031	0	0	0	3.5031	0.0117
40	1	2.5541	0	0	0	2.5541	0.0089
41	1	3.8165	0	0	0	3.8165	0.0096
42	3	2.8481	0.5932	0	0	3.4413	0.0107
43	4	1.082	0.4939	0.3054	0.4313	2.3126	0.0069
44	2	2.4653	0.8598	0	0	3.3251	0.0094
45	1	3.7924	0	0	0	3.7924	0.0104
46	2	0.8548	2.4587	0	0	3.3135	0.0101
47	2	3.5278	0.2277	0	0	3.7555	0.0099
48	1	3.2989	0	0	0	3.2989	0.0101
49	1	2.9311	0	0	0	2.9311	0.009
50	1	2.2792	0	0	0	2.2792	0.0079
51	1	1.5986	0	0	0	1.5986	0.0045
52	1	3.0276	0	0	0	3.0276	0.01
53	1	2.6772	0	0	0	2.6772	0.0092
54	2	1.1925	0.9631	0	0	2.1556	0.0081
55	1	2.7659	0	0	0	2.7659	0.0091
56	1	3.8223	0	0	0	3.8223	0.0103
57	2	1.5294	1.3907	0	0	2.9201	0.0081
58	1	3.6789	0	0	0	3.6789	0.0107
59	1	4.1109	0	0	0	4.1109	0.0116
60	1	4.4066	0	0	0	4.4066	0.0121
61	1	2.9793	0	0	0	2.9793	0.0098

Output of automatic nuclear analysis for the ZT04\_01 SCN SBEM dataset – Contents of *nucleolus\_morphology.txt*, Continued.

Cell #	# of Nucleoli	Volume: Nucleolus #1 ( $\mu\text{m}^3$ )	Volume: Nucleolus #2 ( $\mu\text{m}^3$ )	Volume: Nucleolus #3 ( $\mu\text{m}^3$ )	Volume: Nucleolus #4 ( $\mu\text{m}^3$ )	Total Nucleolar Volume ( $\mu\text{m}^3$ )	Nucleolar Volume Fraction
62	1	3.6055	0	0	0	3.6055	0.0107
63	1	4.0165	0	0	0	4.0165	0.0118
64	2	1.3151	1.508	0	0	2.8231	0.0085
65	1	4.2344	0	0	0	4.2344	0.0116
66	2	1.6483	0.5725	0	0	2.2208	0.0076
67	3	0.8395	1.0996	0.4373	0	2.3764	0.007
68	1	3.0971	0	0	0	3.0971	0.0092
69	2	1.6449	1.8453	0	0	3.4902	0.0101
70	3	1.9681	0.5075	0.4405	0	2.9161	0.0099
71	1	2.7444	0	0	0	2.7444	0.009
72	2	1.5272	0.2569	0	0	1.7841	0.0066
73	1	3.5772	0	0	0	3.5772	0.0102
74	1	3.5818	0	0	0	3.5818	0.0113
75	2	4.2096	0.4054	0	0	4.615	0.0123
76	1	2.2789	0	0	0	2.2789	0.0074
77	2	2.4231	0.7827	0	0	3.2058	0.0101
78	1	2.6452	0	0	0	2.6452	0.0091
79	2	2.4552	1.5774	0	0	4.0326	0.0108
80	2	1.4105	2.0149	0	0	3.4254	0.0103
81	1	3.5928	0	0	0	3.5928	0.0124

Output of automatic nuclear analysis for the ZT04\_01 SCN SBEM dataset – Contents of *dist\_centroid.txt*.

Cell #	Distance: Nucleolus #1 ( $\mu\text{m}$ )	Distance: Nucleolus #2 ( $\mu\text{m}$ )	Distance: Nucleolus #3 ( $\mu\text{m}$ )	Distance: Nucleolus #4 ( $\mu\text{m}$ )
1	7.9436	6.1031	0	0
2	6.8471	2.5216	0	0
3	3.1866	8.7385	0	0
4	7.6548	0	0	0
5	4.7801	0	0	0
6	3.4784	0	0	0
7	5.0477	9.3676	0	0
8	1.3088	4.4262	0	0
9	4.2182	0	0	0

Output of automatic nuclear analysis for the ZT04\_01 SCN SBEM dataset – Contents of *dist\_centroid.txt*, Continued.

<b>Cell #</b>	<b>Distance: Nucleolus #1 (<math>\mu\text{m}</math>)</b>	<b>Distance: Nucleolus #2 (<math>\mu\text{m}</math>)</b>	<b>Distance: Nucleolus #3 (<math>\mu\text{m}</math>)</b>	<b>Distance: Nucleolus #4 (<math>\mu\text{m}</math>)</b>
10	5.0035	7.8152	7.774	0
11	3.966	0	0	0
12	5.6642	0	0	0
13	1.2586	0	0	0
14	3.5343	3.4318	0	0
15	5.0777	0	0	0
16	10.0281	3.7503	8.0568	0
17	1.5045	0	0	0
18	6.2651	5.0259	0	0
19	10.4327	3.8781	0	0
20	1.5227	0	0	0
21	8.3864	4.2999	5.1474	0
22	4.0229	0	0	0
23	3.0527	0	0	0
24	3.0628	0	0	0
25	2.8466	0	0	0
26	2.0245	0	0	0
27	1.5623	0	0	0
28	8.18	2.5233	7.6754	0
29	3.0441	0	0	0
30	7.2927	4.9024	0	0
31	3.8328	12.223	0	0
32	5.0906	6.8841	0	0
33	5.626	10.5948	10.0901	0
34	8.9506	8.6125	0	0
35	5.8478	0	0	0
36	4.3511	0	0	0
37	3.0174	7.6177	0	0
38	3.2767	0	0	0
39	2.3454	0	0	0
40	3.9996	0	0	0
41	4.019	0	0	0
42	4.0502	9.6341	10.07	0
43	6.5722	7.7395	5.5262	11.6972
44	4.2894	3.4747	0	0
45	5.1087	0	0	0
46	5.7078	2.1161	0	0



Output of automatic nuclear analysis for the ZT04\_01 SCN SBEM dataset – Contents of *dist\_centroid.txt*, Continued.

Cell #	Distance: Nucleolus #1 ( $\mu\text{m}$ )	Distance: Nucleolus #2 ( $\mu\text{m}$ )	Distance: Nucleolus #3 ( $\mu\text{m}$ )	Distance: Nucleolus #4 ( $\mu\text{m}$ )
47	4.9322	4.4599	0	0
48	2.6322	0	0	0
49	2.4348	0	0	0
50	2.7891	0	0	0
51	7.8406	0	0	0
52	2.8402	0	0	0
53	2.386	0	0	0
54	5.3376	6.767	0	0
55	4.1016	0	0	0
56	2.959	0	0	0
57	6.0671	4.8251	0	0
58	2.1008	0	0	0
59	1.4003	0	0	0
60	5.3229	0	0	0
61	4.5505	0	0	0
62	4.7134	0	0	0
63	2.5462	0	0	0
64	5.7316	5.0322	0	0
65	3.0135	0	0	0
66	1.3655	9.2155	0	0
67	7.3682	7.8983	7.1588	0
68	4.0141	0	0	0
69	3.4264	2.9334	0	0
70	3.548	3.0666	8.9977	0
71	3.9479	0	0	0
72	5.0125	7.121	0	0
73	4.6562	0	0	0
74	0.8245	0	0	0
75	4.8804	8.2702	0	0
76	7.8593	0	0	0
77	2.9941	5.0876	0	0
78	3.5169	0	0	0
79	6.1666	4.4882	0	0
80	7.5027	5.83	0	0
81	2.2331	0	0	0

Output of automatic nuclear analysis for the ZT04\_01 SCN SBEM dataset – Contents of *dist\_nuclear\_envelope.txt*.

<b>Cell #</b>	<b>Distance: Nucleolus #1 (<math>\mu\text{m}</math>)</b>	<b>Distance: Nucleolus #2 (<math>\mu\text{m}</math>)</b>	<b>Distance: Nucleolus #3 (<math>\mu\text{m}</math>)</b>	<b>Distance: Nucleolus #4 (<math>\mu\text{m}</math>)</b>
1	0	1.2194	0	0
2	0.02	1.0684	0	0
3	0.4894	0.0974	0	0
4	0.0231	0	0	0
5	0.9073	0	0	0
6	0.7257	0	0	0
7	1.3855	0.0654	0	0
8	0.1744	0.0648	0	0
9	0.8158	0	0	0
10	0.8047	0.1014	0.0213	0
11	0.963	0	0	0
12	0.3777	0	0	0
13	1.4539	0	0	0
14	0.9015	0.8656	0	0
15	0	0	0	0
16	0.0433	0.7692	0.1214	0
17	0.8347	0	0	0
18	0.9725	1.3459	0	0
19	0	0.664	0	0
20	0.6991	0	0	0
21	0.2414	0.7908	0.0178	0
22	0.9405	0	0	0
23	0.7386	0	0	0
24	1.1308	0	0	0
25	0.8245	0	0	0
26	0.8742	0	0	0
27	0.7205	0	0	0
28	0.9496	1.176	0.1042	0
29	0.1889	0	0	0
30	0	0.897	0	0
31	1.2875	1.6732	0	0
32	0.138	0.2803	0	0
33	0	0	0	0
34	0.0077	0.0655	0	0
35	0.3478	0	0	0
36	0.9209	0	0	0
37	0	0.8563	0	0

Output of automatic nuclear analysis for the ZT04\_01 SCN SBEM dataset – Contents of *dist\_nuclear\_envelope.txt*, Continued.

<b>Cell #</b>	<b>Distance: Nucleolus #1 (<math>\mu\text{m}</math>)</b>	<b>Distance: Nucleolus #2 (<math>\mu\text{m}</math>)</b>	<b>Distance: Nucleolus #3 (<math>\mu\text{m}</math>)</b>	<b>Distance: Nucleolus #4 (<math>\mu\text{m}</math>)</b>
38	0.6581	0	0	0
39	0.8727	0	0	0
40	0.5974	0	0	0
41	0.547	0	0	0
42	0	0	0	0
43	0.1339	0.1716	0.0054	0.0458
44	1.2674	0.0808	0	0
45	0.7763	0	0	0
46	0	0.979	0	0
47	0.6622	0.1453	0	0
48	1.0309	0	0	0
49	0.752	0	0	0
50	1.3005	0	0	0
51	0.4094	0	0	0
52	1.0623	0	0	0
53	1.2228	0	0	0
54	0.5688	0.0676	0	0
55	1.2784	0	0	0
56	0.7104	0	0	0
57	0.6959	0.143	0	0
58	1.5181	0	0	0
59	1.9465	0	0	0
60	1.0811	0	0	0
61	1.3852	0	0	0
62	0.2054	0	0	0
63	0.292	0	0	0
64	0.0387	0.0703	0	0
65	0.8027	0	0	0
66	0.2548	0.6244	0	0
67	0.2699	0.9479	0.5666	0
68	0.3923	0	0	0
69	1.0019	0.9277	0	0
70	1.0667	0.9677	0.0534	0
71	0.9473	0	0	0
72	0.0063	0	0	0
73	1.0315	0	0	0
74	1.1774	0	0	0

Output of automatic nuclear analysis for the ZT04\_01 SCN SBEM dataset – Contents of *dist\_nuclear\_envelope.txt*, Continued.

<b>Cell #</b>	<b>Distance: Nucleolus #1 (<math>\mu\text{m}</math>)</b>	<b>Distance: Nucleolus #2 (<math>\mu\text{m}</math>)</b>	<b>Distance: Nucleolus #3 (<math>\mu\text{m}</math>)</b>	<b>Distance: Nucleolus #4 (<math>\mu\text{m}</math>)</b>
75	0.7432	0	0	0
76	0	0	0	0
77	1.1646	0.1955	0	0
78	0.7599	0	0	0
79	1.0683	0.1098	0	0
80	0.9879	0.1151	0	0
81	0.6593	0	0	0

## References

- Abbott, A. (2013). Solving the brain. *Nature*, 499, 272-274.
- Abe, T., Takano, K., Suzuki, A., Shimada, Y., Inagaki, M., Sato, N., Obinata, T., and Endo, T. (2004). Myocyte differentiation generates nuclear invaginations traversed by myofibrils associating with sarcomeric protein mRNAs. *The Journal of Cell Science*, 117, 6523-6534.
- Allen, B.A., and Levinthal, C. (1990). Cartos II semi-automated nerve tracing: Three-dimensional reconstruction from serial section micrographs. *Computerized Medical Imaging and Graphics*, 14(5), 319-329.
- Anderson, J.R., Jones, B.W., Watt, C.B., Shaw, M.V., Yang, J.H., Demill, D., Lauritzen, J.S., Lin, Y., Rapp, K.D., Mastronarde, D., Koshevoy, P., Grimm, B., Tasdizen, T., Whitaker, R., and Marc, R.E. (2011). Exploring the retinal connectome. *Molecular Vision*, 17, 355-379.
- Anderson, J.R., Mohamed, S., Grimm, B., Jones, B.W., Koshevoy, P., Tasdizen, T., Whitaker, R., Marc, R.E. (2010). The Viking viewer for connectomics: Scalable multi-user annotation and summarization of large volume data sets. *The Journal of Microscopy*, 241(1), 1328.
- 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. *Medical Image Analysis*, 16(4), 796-805.
- Anttonen, T., Kirjavainen, A., Belevich, I., Laos, M., Richardson, W.D., Jokitalo, E., Brakebusch, C., and Pirvola, U. (2012). CdC42-dependent structural development of auditory supporting cells is required for wound healing at adulthood. *Scientific Reports*, 2, 978.
- Arkill, K.P., Qvortrup, K., Starborg, T., Mantell, J.M., Knupp, C., Michel, C.C., Harper, S.J., Salmon, A.H.J., Squire, J.M., Bates, D.O., and Neal, C.R. (2014). Resolution of the three dimensional structure of components of the glomerular filtration barrier. *BMC Nephrology*, 15(1), 24.
- Armer, H.E.J., Mariggi, G., Png, K.M.Y., Genoud, C., Monteith, A.G., Bushby, A.J., Gerhardt, H., and Collinson, L.M. (2009). Imaging transient blood vessel fusion events in zebrafish by correlative volume electron microscopy. *PLoS one*, 4(11), e7716.
- Armstrong, W.E., and Hatton, G.I. (1978). Morphological changes in the rat supraoptic and paraventricular nuclei during the diurnal cycle. *Brain Research*, 157, 407-413.
- Austin, J.R., Segui-Simarro, J.M., and Staehelin, L.A. (2005). Quantitative analysis of changes in spatial distribution and plus-end geometry of microtubules involved in plant-cell cytokinesis. *The Journal of Cell Biology*, 118, 3895-3903.

- Bai, X., Latecki, L.J., and Liu, W.-Y. (2007). Skeleton pruning by contour partitioning with discrete curve evolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(3), 449-462.
- Baker, T.S., and Johnson, J.E. (1996). Low resolution meets high: Towards a resolution continuum from cells to atoms. *Current Opinion in Structural Biology*, 6, 585-594.
- Baldi, P., Brunak, S., Chauvin, Y., Andersen, C.A., and Nielsen, H. (2000). Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics*, 16(5), 412-424.
- Ball, P. (2014). Crowd-sourcing: Strength in numbers. *Nature*, 506(7489), 422-423.
- Balsalobre, A., Damiola, F., and Schibler, U. (1998). A serum shock induces circadian gene expression in mammalian tissue culture cells. *Cell*, 93, 929-937.
- Bang, B.G., and Bang, F.B. (1957). Graphic reconstruction of the third dimension from serial electron microphotographs. *The Journal of Ultrastructure Research*, 1, 138-146.
- Barnum, C.P., Jardetzky, C.D., and Halberg, F. (1958). Time relations among metabolic and morphologic 24 hour changes in mouse liver. *American Journal of Physiology*, 195, 301-310.
- Becker, U.G., and Vollrath, L. (1983). 24-hour variation of pineal gland volume, pinealocytes nuclear volume, and mitotic activity in male Sprague-Dawley rats. *The Journal of Neural Transmission*, 56, 211-221.
- Becquet, D., Girardet, C., Guillaumond, F., Francois-Bellan, A.-M., and Bosler, O. (2008). Ultrastructural plasticity in the rat suprachiasmatic nucleus. Possible involvement in clock entrainment. *Glia*, 56, 294-305.
- Bell-Pedersen, D., Cassone, V.M., Earnest, D.J., Golden, S.S., Hardin, P.E., Thomas, T.L., and Zoran, M.J. (2005). Circadian rhythms from multiple oscillators: lessons from diverse organisms. *Nature Reviews Drug Discovery*, 6, 554-556.
- Benson, B., and Krasovich, M. (1977). Circadian rhythm in the number of granulated vesicles in the pinealocytes of mice. *Cell and Tissue Research*, 184, 499-506.
- Berlanga, M.L., Phan, S., Bushong, E.A., Wu, S., Kwon, O., Phung, B.S., Lamont, S., Terada, M., Tasdizen, T., Martone, M.E., and Ellisman, M.H. (2011). Three-dimensional reconstruction of serial mouse brain sections: Solution for flattening high-resolution large-scale mosaics. *Frontiers in Neuroanatomy*, 5, 17.
- Besl, P.J., and McKay, N.D. (1992). A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14, 239-256.
- Bessone, R., and Seite, R. (1985). Daily fluctuations of nucleoli in neurosecretory cells of the rat supraoptic nucleus. An ultrastructural and stereological study. *Cell and Tissue Research*, 240, 393-396.

- Betzig, B., Patterson, G.H., Sougrat, R., Lindwasser, O.W., Olenych, S., Bonifacino, J.S., Davidson, M.W., Lippincott-Schwartz, J., Hess, H.F. (2006). Imaging intracellular fluorescent proteins at nanometer resolution. *Science*, 15, 313(5793), 1642-1645.
- Blazquez, J., Pastor, F., Amat, P., Pelaez, B., Sanchez, A., and Amat-Peral, G. (1995). Giant granular filamentous bodies in the cytoplasm of arcuate nucleus neurons of castrated rats. *Histology and Histopathology*, 10, 385-392.
- Boassa, D., Berlanga, M.L., Yang, M.A., Terada, M., Hu, J., Bushong, E.A., Hwang, M., Masliah, E., George, J.M., and Ellisman, M.H. (2013). Mapping the subcellular distribution of  $\alpha$ -synuclein in neurons using genetically encoded probes for correlated light and electron microscopy: Implications for Parkinson's disease pathogenesis. *The Journal of Neuroscience*, 33(6), 2605-2615.
- Bock, D.D., Lee, W.-C.A., Kerlin, A.M., Andermann, M.L., Hood, G., Wetzel, A.W., Yurgenson, S., Soucy, E.R., Kim, H.S., and Reid, R.C. (2011). Network anatomy and in vivo physiology of visual cortical neurons. *Nature*, 471, 177-182.
- Bohórquez, D.V., Samsa, L.A., Roholt, A., Medicetty, S., Chandra, R., and Liddle, R.A. (2014). An enteroendocrine cell-enteric glia connection revealed by 3D electron microscopy. *PLoS one*, 9(2), e89881.
- Bonnet, N. (2004). Some trends in microscope image processing. *Micron*, 35, 635-653.
- Bossy-Wetzel, E., Barsoum, M.J., Godzik, A., Schwarzenbacher, R., and Lipton, S.A. (2003). Mitochondrial fission in apoptosis, neurodegeneration and aging. *Current Opinion in Cell Biology*, 15(6), 706-716.
- Braverman, I.M. and Keh-Yen, M.D. (1983). Ultrastructure and three-dimensional reconstruction of several macular and popular telangiectases. *The Journal of Investigative Dermatology*, 81, 489-497.
- Briggman, K.L., and Bock, D.D. (2012). Volume electron microscopy for neuronal circuit reconstruction. *Current Opinion in Neurobiology*, 22, 154-161.
- Briggman, K.L., and Denk, W. (2006). Towards neural circuit reconstruction with volume electron microscopy techniques. *Current Opinion in Neurobiology*, 16, 562-570.
- Briggman, K.L., Helmstaedter, M., and Denk, W. (2011). Wiring specific in the direction-selectivity circuit of the retina. *Nature*, 471, 183-188.
- Bron, C., Sadlo, F., Szekely, G., Neuenschwander, W., Keubler, O., and Schuepbach, H. (1994). "Segmentation and visualization of membranes and intracellular organelles contours in 3D electron microscopy," in *Visualization in Biomedical Computing 1994*, pp. 706-714. International Society for Optics and Photonics.
- Buck, T.E., Li, J., Rohde, G.K., and Murphy, R.F. (2012). Toward the virtual cell: Automated approaches to building models of subcellular organization "learned" from microscopy images. *BioEssays*, 34, 791-799.

- Buckman, J.F., Hernández, H., Kress, G.J., Votyakova, T.V., Pal, S., and Reynolds, I.J. (2001). MitoTracker labeling in primary neuronal and astrocytic cultures: influence of mitochondrial membrane potential and oxidants. *The Journal of Neuroscience Methods*, 165-176.
- Bushby, A.J., Png, K.M.Y., Young, R.D., Pinali, C., Knupp, C., and Quantock, A.J. (2011). Imaging three-dimensional tissue architectures by focused ion beam scanning electron microscopy. *Nature Protocols*, 6(6), 845-858.
- Cai, D., Cohen, K.B., Luo, T., Lichtman, J.W., and Sanes, J.R. (2013). Improved tools for the Brainbow toolbox. *Nature Methods*, 10, 540-547.
- Cajal, S.R. (1906). The structure and connexions of neurons. Nobel lecture, Dec. 12.
- Capell, B.C., and Collins, F.S. (2006). Human laminopathies: nuclei gone genetically awry. *Nature Reviews Genetics*, 7, 940-952.
- Cardona, A. (2013). Towards semi-automatic reconstruction of neural circuits. *Neuroinformatics*, 11, 31-33.
- Cardona, A., Saalfeld, S., Preibisch, S., Schmid, B., Cheng, A., Pulokas, J., Tomancak, P. and Hartenstein, V. (2010). An integrated micro- and macroarchitectural analysis of the Drosophila brain by computer-assisted serial section electron microscopy. *PLoS Biology*, 8.
- 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(6), e38011.
- Carlsson, I., Terzopoulos, D., and Harris, K.M. (1994). Computer-assisted registration, segmentation, and 3D reconstruction from images of neuronal tissue sections. *IEEE Transactions on Medical Imaging*, 13(2), 351-362.
- Carpenter, A.E., Jones, T.R., Lamprecht, M.R., Clarke, C., Kang, I.H., Friman, O., Guertin, D.A., Chang, J.H., Lindquist, R.A., Moffat, J., Golland, P., and Sabatini, D.M. (2006). CellProfiler: Image analysis software for identifying and quantifying cell phenotypes. *Genome Biology*, 7(1), R100.
- Chalfie, M., Sulston, J.E., White, J.G., Southgate, E., Thomson, J.N., and Brenner, S. (1985). The neural circuit for touch sensitivity in *Caenorhabditis elegans*. *The Journal of Neuroscience*, 5(4), 956-964.
- Chan, T.F., and Vese, L.A. (2001). Active contours without edges. *IEEE Transactions on Image Processing*, 10(2), 266-277.
- Chatzis, V., and Pitas, I. (2000). Interpolation of 3-D binary images based on morphological skeletonization. *IEEE Transactions on Medical Imaging*, 19(7):699-710.



- Chklovskii, D.B., Vitaladevuni, S., and Scheffer, L.K. (2010). Semi-automated reconstruction of neural circuits using electron microscopy. *Current Opinion in Neurobiology*, 20, 667-675.
- Choi, S., Wang, W., Ribeiro, A.J.S., Kalinowski, A., Gregg, S.Q., Opresko, P.L., Niedernhofer, L.J., Rohde, G.K., and Dahl, K.N. (2011). Computational image analysis of nuclear morphology associated with various nuclear-specific aging disorders. *Nucleus*, 2(6), 1-10.
- Chow, S.K., Hakozaki, H., Price, D.L., Maclean, N.A.B., Deerinck, T.J., Bouwer, J.C., Martone, M.E., Peltier, S.T., and Ellisman, M.H. (2006). Automated microscopy system for mosaic acquisition and processing. *The Journal of Microscopy*, 222(2), 76-84.
- Chung, K., and Deisseroth, K. (2013). CLARITY for mapping the nervous system. *Nature Methods*, 10(6), 508-513.
- Chung, W.-S., Clarke, L.E., Wang, G.X., Stafford, B.K., Sher, A., Chakraborty, C., Joung, J., Foo, L.C., Thompson, A., Chen, C., Smith, S.J., and Barres, B.A. (2013). Astrocytes mediate synapse elimination through MEGF10 and MERTK pathways. *Nature*, 504(7480), 394-400.
- Churas, C., Lin, A.W., Grethe, J.S., and Ellisman, M.H. (2013). PANFISH: A multi-cluster submission system. *XSEDE13*, San Diego, California, U.S.A.
- Cioce, M., and Lamond, A.I. (2005). Cajal bodies: a long history of discovery. *Annual Review of Cell and Developmental Biology*, 21, 105-131.
- Clubb, B.H., and Locke, M. (1998). 3T3 cells have nuclear invaginations containing F-actin. *Tissue & Cell*, 30(6), 684-691.
- Collings, D.A., Carter, C.N., Rink, J.C., Scott, A.C., Wyatt, S.E., and Allen, N.S. (2000). Plant nuclei can contain extensive grooves and invaginations. *The Plant Cell*, 12, 2425-2439.
- Coltuc, D., Bolon, P., and Chassery, J.M. (2006). Exact histogram specification. *IEEE Transactions on Image Processing*, 15(5), 1143-1152.
- Cowan, W.M., and Wann, D.F. (1973). A computer system for the measurement of cell and nuclear sizes. *The Journal of Microscopy*, 99(3), 331-348.
- Cremer, T., and Cremer, C. (2001). Chromosome territories, nuclear architecture and gene regulation in mammalian cells. *Nature Reviews Genetics*, 2, 292-301.
- Crisp, M., Liu, Q., Roux, K., Rattner, J.B., Shanahan, C., Burke, B., Stahl, P.D., and Hodzic, D. (2006). Coupling of the nucleus and cytoplasm: Role of the LINC complex. *The Journal of Cell Biology*, 172(1), 41-53.
- Crowther, R.A., Henderson, R., and Smith, J.M. (1996). MRC image processing programs. *The Journal of Structural Biology*, 116(1), 9-16.

- Dani, A., Huang, B., Bergan, J., Dulac, C., Zhuang, X. (2011). Superresolution imaging of chemical synapses in the brain. *Neuron*, 68, 843-856.
- Dardick, I., Sinnott, N.M., Hall, R., Bajenko-Carr, A., and Setterfield, G. (1982). Nuclear morphology and morphometry of B-lymphocyte transformation. *The American Journal of Pathology*, 111(1), 35-49.
- Dayal, A., and Hill, D.L. (2014). Image J software designed to quantify multiple labels in sectioned tissue from confocal stacks in large experimental datasets. Program No. 854.01. 2014 Neuroscience Meeting Planner. Washington, D.C.: *Society for Neuroscience*, 2014. Online.
- de Brito, O.M., and Scorrano, L. (2010). An intimate liaison: spatial organization of the endoplasmic reticulum-mitochondria relationship. *The EMBO Journal*, 29, 2715-2723.
- Deerinck, T.J., Martone, M.E., Lev-Ram, V., Green, D.P., Tsien, R.Y., Spector, D.L., Huang, S., and Ellisman, M.H. (1994). Fluorescence photooxidation with eosin: A method for high resolution immunolocalization and in situ hybridization detection for light and electron microscopy. *The Journal of Cell Biology*, 126(4), 901.
- Deerinck, T.J., Bushong, E.A., Lev-Ram, V., Shu, X., Tsien, R.Y., and Ellisman, M.H. (2010). Enhancing serial block-face scanning electron microscopy to enable high resolution 3-D nanohistology of cells and tissues. *Microscopy and Microanalysis*, 16(S2), 1138-1139.
- Denk, W., Strickler, J.H., and Webb, W.W. (1990). Two-photon laser scanning fluorescence microscopy. *Science*, 248(4951), 73-76.
- Denk, W., and Horstmann, H. (2004). Serial block-face scanning electron microscopy to reconstruct three-dimensional tissue nanostructure. *PLoS Biology*, 2(11), 1900-1909.
- DeRosier, D., Stokes, D.L., and Darst, S.A. (1999). Averaging data derived from images of helical structures with different symmetries. *The Journal of Molecular Biology*, 289, 159-165.
- DeRosier, D.J., and Klug, A. (1968). Reconstruction of three-dimensional structures from electron micrographs. *Nature*, 217, 130-134.
- Diehl, B.J.M. (1981). Time-related changes in size of nuclei of pinealocytes in rats. *Cell and Tissue Research*, 218, 427-438.
- Diehl, B.J.M., Heidbüchel, U., Welker, H.A., and Vollrath, L. (1984). Day/night changes of pineal gland volume and pinealocyte nuclear size assessed over 10 consecutive days. *The Journal of Neural Transmission*, 60, 19-29.
- Dolci, C., Vizzotto, L., Morini, M., Ferrari, A., Carandente, F., and Miani, A. (1990). "Structural and ultrastructural circadian features in rat exocrine pancreas," in *Chronobiology: Its Role in Clinical Medicine, General Biology, and Agriculture, Part A*, pp. 235-241. Wiley-Liss, Inc., Wilmington, Delaware, U.S.A.

Donald, A.M. (2003). The use of environmental scanning electron microscopy for imaging wet and insulating materials. *Nature Materials*, 2, 511-516.

Dow, E., Buckley, Y., Berning, M., Bocklisch, T., Braunlein, D., Herold, T., Rzepka, N., Werkmeister, T., and Helmstaeder, M. (2014). Project Brainflight: Scaling connectomics reconstruction via lay-audience targeted image analysis. Program No. 98.05. 2014 Neuroscience Meeting Planner. Washington, D.C.: *Society for Neuroscience*, 2014. Online.

Eling, W. (1967). "The circadian rhythm of nucleic acids," in *The Cellular Aspects of Biorhythms*, pp. 105-114. Springer Berlin Heidelberg.

Ellisman, M.H., Boassa, D., Nguyen, P., Wan, X., Lawrence, A., Lanman, J., and Phan, S. (2014). Automated procedures for the alignment and reconstruction of multiple tilt electron microscopic tomography data. *Microscopy and Microanalysis*, 20(S3), 1258-1259.

Ewald, A.J., Huebner, R.J., Palsdottir, H., Lee, J.K., Perez, M.J., Jorgens, D.M., Tauscher, A.N., Cheung, K.J., Werb, Z., and Auer, M. (2012). Mammary collective cell migration involves transient loss of epithelial features and individual cell migration within the epithelium. *The Journal of Cell Science*, 125, 2638-2654.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, 27, 861-874.

Feierbach, B., Piccinotti, S., Bisher, M., Denk, W., and Enquist, L.W. (2006). Alpha-herpesvirus infection induces the formation of nuclear actin filaments. *PLoS pathogens*, 2(8), e85.

Fernandez, B., Suarez, I., and Gianonatti, C. (1983). Fine structure of astrocytic mitochondria in the hypothalamus of the hamster. *The Journal of Anatomy*, 137(Pt 3), 483-488.

Fiala, J.C. (2005). Reconstruct: A free editor for serial section microscopy. *The Journal of Microscopy*, 218(1), 52-61.

Fiala, J.C., Feinberg, M., Popov, V., and Harris, K.M. (1998). Synaptogenesis via dendritic filopodia in developing hippocampal area CA1. *The Journal of Neuroscience*, 18, 8900-8911

Folk, M., Cheng, A., and Yates, K. (1999). "HDF5: A file format and I/O library for high performance computing applications," in *Proceedings of Supercomputing*, 99.

Fox, C.A., Rafols, J.A., and Cowan, W.M. (1975). Computer measurements of axis cylinder diameters of radial fibers and "comb" bundle fibers. *The Journal of Comparative Neurology*, 159(2), 201-223.

Francis, N.R., Sosinsky, G.E., Thomas, D., and DeRosier, D.J. (1994). Isolation, characterization and structure of bacterial flagellar motors containing the switch complex. *The Journal of Molecular Biology*, 235, 1261-1270.

Frank, J. (1990). Classification of macromolecular assemblies studied as single particles. *Quarterly Reviews of Biophysics*, 23, 281-329.

Frank, J. (2002). Single-particle imaging of macromolecules by cryo-electron microscopy. *Annual Review of Biophysics and Biomolecular Structure*, 31, 303-319.

Frank, J. (2008). *Electron Tomography: Methods for Three-Dimensional Visualization of Structures in the Cell*. 2<sup>nd</sup> edition. Springer New York.

Frey, T.G., Perkins, G.A., and Ellisman, M.H. (2005). Electron tomography of membrane-bound organelles. *Annual Review of Biophysics and Biomolecular Structure*. 35, 199-224.

Fricker, M., Hollinshead, M., White, N., and Vaux, D. (1997). Interphase nuclei of many mammalian cell types contain deep, dynamic, tubular membrane-bound invaginations of the nuclear envelope. *The Journal of Cell Biology*, 136(3), 531-544.

Fuchs, P.A., Lehar, M., and Hiel, H. (2014). Ultrastructure of cisternal synapses on outer hair cells of the mouse cochlea. *The Journal of Comparative Neurology*, 522(3), 717-729.

Fujinaga, R., Yanai, A., Nakatsuka, H., Yoshida, K., Takeshita, Y., Uozumi, K., Zhao, C., Hirata, K., Kokubu, K., Nagano, M., and Shinoda, K. (2007). Anti-human placental antigen complex X-P2 (hPAX-P2) anti-serum recognizes C-terminus of huntingtin-associated protein 1A common to 1B as a determinant marker for the stigmoid body. *Histochem Cell Biol*, 128, 335-348.

Gaietta, G., Deerinck, T.J., Adams, S.R., Bouwer, J., Tour, O., Laird, D.W., Soinsky, G.E., Tsien, R.Y., and Ellisman, M.H. (2002). Multicolor and electron microscopic imaging of connexin trafficking. *Science*, 296, 503-507.

Gan, W.-B., Grutzendler, J., Wong, W.T., Wong, R.O.L., Lichtman, J.W. (2000). Multicolor "DiOlistic" labeling of the nervous system using lipophilic dye combinations. *Neuron*, 27(2), 219-225.

Gan, L., and Jensen, G.J. (2012). Electron tomography of cells. *Quarterly Reviews of Biophysics*, 45(1), 27-56.

Gay, H., and Anderson, T.F. (1954). Serial sections for electron microscopy. *Science*, 120(3130), 1071-1073.

Gerasimov, A.V., Logvinov, S.V., and Kosyuchenko, V.P. (2010). Morphological changes in the pineal gland of rats under conditions of long-term exposure to bright light. *Bulletin of Experimental Biology and Medicine*, 150(1), 86-88.

Gibeaux, R., Hoepfner, D., Schlatter, I., Antony, C., and Philippsen, P. (2013). Organization of organelles within hyphae of *Ashbya gossypii* revealed by electron tomography. *Eukaryotic Cell*, 12(11), 1423.

Giepmans, B.N.G., Deerinck, T.J., Smar, B.L., Jones, Y.Z., and Ellisman, M.H. (2005). Correlated light and electron microscopic imaging of multiple endogenous proteins using quantum dots. *Nature Methods*, 2(10), 743-749.

Giepmans, B.N.G., Adams, S.R., Ellisman, M.H., and Tsien, R.Y. (2006). The fluorescent toolbox for assessing protein location and function. *Science*, 312, 217-224.

Girardet, C., Lebrun, B., Cabirol-Pol, M.-J., Tardivel, C., Francois-Bellan, A.-M., Becquet, D., and Bosler, O. (2014). Brain-derived neurotrophic factor/TrkB signaling regulates daily astroglial plasticity in the suprachiasmatic nucleus: Electron-microscopic evidence in mouse. *Glia*, 61, 1172-1177.

Giuly, R.J., Kim, K.-Y., and Ellisman, M.H. (2013). DP2: Distributed 3D image segmentation using micro-labor workforce. *Bioinformatics*, 29(10), 1359-1360.

Giuly, R.J., Martone, M.E., and Ellisman, M.H. (2012). Method: Automatic segmentation of mitochondria utilizing patch classification, contour pair classification, and automatically seeded level sets. *BMC Bioinformatics*, 13(29).

Glaeser, R.M. (1999). Review: Electron crystallography: Present excitement, a nod to the past, anticipating the future. *The Journal of Structural Biology*, 128(1), 3-14.

Golgi, C. (1906). The neuron doctrine – theory and facts. Nobel lecture, Dec. 11.

Goodchild, R.E., Kim, C.E., and Dauer, W.T. (2005). Loss of the dystonia-associated protein torsinA selectively disrupts the neuronal nuclear envelope. *Neuron*, 48(6), 923-932.

Groh, V., and von Mayersbach, H. (1981). Enzymatic and functional heterogeneity of lysosomes. *Cell and Tissue Research*, 214, 613-621.

Grynkiewicz, G., Poenie, M. and Tsien, R.Y. (1985). A new generation of Ca<sup>2+</sup> indicators with greatly improved fluorescence properties. *The Journal of Biological Chemistry*, 260, 3440-3450.

Güldner, F.-H. (1976). Synaptology of the rat suprachiasmatic nucleus. *Cell and Tissue Research*, 165, 509-544.

Gustafsson, M.G.L. (2000). Surpassing the lateral resolution limit by a factor of two using structured illumination microscopy. *The Journal of Microscopy*, 195, 10-16.

Gutkunst, C., Li, S., Yi, H., Ferrante, R.J., Li, X., and Hersch, S.M. (1998). The cellular and subcellular localization of huntingtin-associated protein 1 (HAP1): Comparison with huntingtin in rat and human. *The Journal of Neuroscience*, 18(19), 7674-86.

Gutkunst, C., Torre, E.R., Sheng, Z., Yi, H., Coleman, S.H., Riedel, I.B., and Bujo, H., (2003). Stigmoid bodies contain type I receptor proteins SorLA/LR11 and sortilin: New perspectives on their function. *The Journal of Histochemistry & Cytochemistry*, 51(6), 841-52.

- Hall, D.H., and Russell, R.L. (1991). The posterior nervous system of the nematode *Caenorhabditis elegans*: Serial reconstruction of identified neurons and complete pattern of synaptic interactions. *The Journal of Neuroscience*, 11(1), 1-22.
- Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I.H. (2009). The WEKA data mining software: An update. *ACM SIGKDD Explorations Newsletter*, 11(1), 10-18.
- Handwerker, K.E., and Gall, J.G. (2006). Subnuclear organelles: new insights into form and function. *Trends in Cell Biology*, 16(1), 19-26.
- Hardeland, R., Hohmann, D., and Rensing, L. (1973). The rhythmic organization of rodent liver. A review. *The Journal of Interdisciplinary Cycle Research*, 4(2), 89-118.
- Harlow, M.L., Ress, D., Stoscheck, A., Marshall, R.M., and McMahan, U.J. (2001). The architecture of active zone material at the frog's neuromuscular junction. *Nature*, 409, 479-484.
- Harris, K.M., and Stevens, J.K. (1988). Dendritic spines of rat cerebellar Purkinje cells: serial electron microscopy with reference to their biophysical characteristics. *The Journal of Neuroscience*, 8(12), 4455-4489.
- Harris, K.M. (1999). Structure, development, and plasticity of dendritic spines. *Current Opinion in Neurobiology*, 9, 343-348.
- Hatori, M., Vollmers, C., Zarrinpar, A., DiTacchio, L., Bushong, E.A., Gill, S., Leblanc, M., Chaix, A., Joens, M., Fitzpatrick, J.A.J., Ellisman, M.H., and Panda, S. (2012). Time-restricted feeding without reducing caloric intake prevents metabolic diseases in mice fed a high-fat diet. *Cell Metabolism*, 15(6), 848-860.
- Hattar, S., Liao, H.-W., Takao, M., Berson, D.M., and Yau, K.-W. (2002). Melanopsin-containing retinal ganglion cells: architecture, projections, and intrinsic photosensitivity. *Science*, 295(5557), 1065-1070.
- Hayworth, K.J., Kasthuri, N., Schalek, R., and Lichtman, J.W. (2006). Automating the collection of ultrathin serial sections for large volume TEM reconstructions. *Microscopy and Microanalysis*, 12(Supp 2), 86-87.
- Hayworth, K.J., Morgan, J.L., Schalek, R., Berger, D.R., Hildebrand, D.G.C., and Lichtman, J.W. (2014). Imaging ATUM ultrathin section libraries with WaferMapper: a multi-scale approach to EM reconstruction of neural circuits. *Frontiers in Neural Circuits*, 8, 68.
- Hellman, B., and Hellerström, C. (1959). Diurnal changes in the function of the pancreatic islets of rats as indicated by nuclear size in the islet cells. *Acta Endocrinologica*, 31, 267-281.

- Helmstaedter, M., Briggman, K.L., Turaga, S.C., Jain, V., Seung, H.S., and Denk, W. (2013). Connectomic reconstruction of the inner plexiform layer in the mouse retina. *Nature*, 500, 168-174.
- Herms, A., Bosch, M., Ariotti, N., Reddy, B.J.N., Fajardo, A., Fernandez-Vidal, A., Alvarez-Guaita, A., Fernandez-Rojo, M.A., Rentero, C., Tebar, F., Enrich, C., Geli, M.-I., Parton, R.G., Gross, S.P., and Pol, A. (2013). Cell-to-cell heterogeneity in lipid droplets suggests a mechanism to reduce lipotoxicity. *Current Biology*, 23(15), 1489-1496.
- Hessler, D., Young, S.J., Carragher, B.O., Martone, M.E., Lamont, S., Whittaker, M., Milligan, R.A., Masliah, E., Hinshaw, J.E., and Ellisman, M.H. (1992). Programs for visualization in three-dimensional microscopy. *Neuroimage*, 1, 55-67.
- Hetman, M., and Pietrzak, M. (2012). Emerging roles of the neuronal nucleolus. *Trends in Neurosciences*, 35(5), 305-314.
- Heymann, J.A.W., Hayles, M., Gestmann, I., Giannuzzi, L.A., Lich, B., and Subramaniam, S. (2006). Site-specific 3D imaging of cells and tissues with a dual beam microscope. *The Journal of Structural Biology*, 155, 63-73.
- Heymann, J.B. (2001). Bsoft: Image and molecular processing in electron microscopy. *The Journal of Structural Biology*, 133(2), 156-169.
- Hindelang-Gertner, C., Stoeckel, M.E., Porte, A., Dellmann, H.D., and Madarasz, B. (1974). Nematosomes or nucleolus-like bodies in hypothalamic neurons, the subfornical organ and adenohypophysial cells of the rat. *Cell and Tissue Research*, 155, 211-219.
- Hira, Y., Sakai, Y., and Matsushima, S. (1989). Comparisons of sizes of pinealocyte nuclei and pinealocytes in young and adult Chinese hamsters (*Cricetulus griseus*) under different photoperiod conditions. *The Journal of Pineal Research*, 7, 411-418.
- Hoelz, A., Debler, E.W., and Blobel, G. (2011). The structure of the nuclear pore complex. *Annual Review in Biochemistry*, 80, 613-43.
- Höhne, K.H., Bomans, M., Pommert, A., Riemer, M., Schiers, C., Tiede, U., and Wiebecke, G. (1990). 3D visualization of tomographic volume data using the generalized voxel model. *The Visual Computer*, 6, 28-36.
- Hoffman, H.-P., and Avers, C.J. (1973). Mitochondrion of yeast: Ultrastructural evidence for one giant, branched organelle per cell. *Science*, 181(4101), 749-751.
- Holcomb, P.S., Hoffpauir, B.K., Hoyson, M.C., Jackson, D.R., Deerinck, T.J., Marrs, G.S., Dehoff, M., Wu, J., Ellisman, M.H., and Spirou, G.A. (2013). Synaptic inputs compete during rapid formation of the calyx of Held: A new model system for neural development. *The Journal of Neuroscience*, 33(32), 12954-12969.
- Höög, J.L., Schwartz, C., Noon, A.T., O'Toole, E.T., Mastronarde, D.N., McIntosh, J.R., and Antony, C. (2007). Organization of interphase microtubules in fission yeast analyzed by electron tomography. *Developmental Cell*, 12, 349-361.

- Hoppa, M.B., Jones, E., Karanauskaite, J., Ramracheya, R., Braun, M., Collins, S.C., Zhang, Q., Clark, A., Eliasson, L., Genoud, C., MacDonald, P.E., Monteith, A.G., Barg, S., Galvanovskis, J., and Rorsman, P. (2012). Multivesicular exocytosis in rat pancreatic beta cells. *Diabetologia*, 55, 1001-1012.
- Horstmann, H., Körber, C., Sätzler, K., Aydin, D., and Kuner, T. (2012). Serial section scanning electron microscopy (S<sup>3</sup>EM) on silicon wafers for ultra-structural volume imaging of cells and tissues. *PLoS one*, 7(4), e35172.
- Hu, W., Haamedi, N., Lee, J., Kinoshita, T., and Ohnuma, S. (2013). The structure and development of *Xenopus laevis* cornea. *Experimental Eye Research*, 116, 109-128.
- Huang, S., Deerinck, T.J., Ellisman, M.H., and Spector, D.L. (1997). The dynamic organization of the perinucleolar compartment in the cell nucleus. *The Journal of Cell Biology*, 137(5), 965-974.
- Hughes, M.E., DiTacchio, L., Hayes, K.R., Vollmers, C., Pulivarthy, S., Baggs, J.E., Panda, S., and Hogenesch, J.B. (2009). Harmonics of circadian gene transcription in mammals. *PLoS Genetics*, 5, e10000442.
- Hundahl, C.A., Hannibal, J., Fahrenkrug, J., Dewilde, S., and Hay-Schmidt, A. (2010). Neuroglobin expression in the rat suprachiasmatic nucleus: colocalization, innervation, and response to light. *The Journal of Comparative Neurology*, 518, 156-1569.
- Hunter, D.A., Moradzadeh, A., Whitlock, E.L., Brenner, M.J., Myckatyn, T.M., Wei, C.H., Tung, T.H.H., and Mackinnon, S. (2007). Binary imaging analysis for comprehensive quantitative assessment of peripheral nerve. *The Journal of Neuroscience Methods*, 166(1), 116-124.
- Ingber, D.E. (1997). Tensegrity: The architectural basis of cellular mechanotransduction. *Annual Review of Physiology*, 59, 575-599.
- Insel, T.R., Landis, S.C., and Collins, F.S. (2013). The NIH BRAIN Initiative. *Science*, 340, 687-688.
- Ishii, Y., Hasegawa, S., and Uchiyama, Y. (1989). Twenty-four-hour variations in subcellular structures of rat type II alveolar epithelial cells. *Cell and Tissue Research*, 256, 347-353.
- Jaume, S., Knobe, K., Newton, R.R., Schlimbach, F., Blower, M., and Reid, R.C. (2012). A multiscale parallel computing architecture for automated segmentation of the brain connectome. *IEEE Transactions on Biomedical Engineering*, 59(1), 35-38.
- Jeong, W.-K., Beyer, J., Hadwiger, M., Vazquez, A., Pfister, H., and Whitaker, R.T. (2009). Scalable and interactive segmentation and visualization of neural processes in EM datasets. *IEEE Transactions on Visualization and Computer Graphics*, 15(6), 1505-1514.



- Jorstad, A., Nigro, B., Cali, C., Wawrzyniak, M., Fua, P., Knott, G. (2014). NeuroMorph: A toolset for the morphometric analysis and visualization of 3D models derived from electron microscopy image stacks. *Neuroinformatics*, 1-10.
- Jurrus, E., Hardy, M., Tasdizen, T., Fletcher, P., Koshevoy, P. Chien, C.B., Denk, W., and Whitaker, R. (2009). Axon tracking in serial block-face scanning electron microscopy. *Medical Image Analysis*, 13, 180-188.
- Kalson, N.S., Holmes, D.F., Herchenhan, A., Lu, Y., Starborg, T., and Kadler, K.E. (2011). Slow stretching that mimics embryonic growth rate stimulates structural and mechanical development of tendon-like tissue in vitro. *Developmental Dynamics*, 240(11), 2520-2528.
- Kalson, N.S., Starborg, T., Lu, Y., Mironov, A., Humphries, S.M., Holmes, D.F., and Kadler, K.E. (2013). Nonmuscle myosin II powered transport of newly formed collagen fibrils at the plasma membrane. *Proceedings of the National Academy of Sciences U.S.A.*, 110(49), E4743-E4752.
- Kang, B.-H., and Staehelin, L.A. (2008). ER-to-Golgi transport by COPII vesicles in *Arabidopsis* involves a ribosome-excluding scaffold that is transferred with the vesicles to the Golgi matrix. *Protoplasma*, 234, 51-64.
- Kapur, J.N., Sahoo, P.K., and Wong, A.C.K. (1985). A new method for gray-level picture thresholding using the entropy of the histogram. *Graphical Models and Image Processing*, 29(3), 273-285.
- Kapuscinski, J. (1995). DAPI: A DNA-specific fluorescent probe. *Biotechnic & Histochemistry*, 70(5), 220-233.
- Karasek, M., Stankov, B., Lucini, V., Scaglione, F., Esposti, G., Mariani, M., and Fraschini, F. (1990). Comparison of the rat pinealocyte ultrastructure with melatonin concentrations during daytime and at night. *The Journal of Pineal Research*, 9, 251-257.
- Karlsson, U. (1966). Three-dimensional studies of neurons in the lateral geniculate nucleus of the rat. I. Organelle organization in the perikaryon and its proximal branches. *The Journal of Ultrastructure Research*, 16, 429-481.
- Karlsson, U., Andersson-Cedergren, E., and Ottoson, D. (1966). Cellular organization of the frog muscle spindle as revealed by serial sections for electron microscopy. *The Journal of Ultrastructure Research*, 14, 1-35.
- Kass, M., Witkin, A., Terzopoulos, D. (1988). Snakes: Active contour models. *The International Journal of Computer Vision*, 1(4), 321-331.
- Keeley, P.W., Luna, G., Fariss, R.N., Skyles, K.A., Madsen, N.R., Raven, M.A., Poche, R.A., Swindell, E.C., Jamrich, M., Oh, E.C., Swaroop, A., Fisher, S.K., and Reese, B.E. (2013). Development and plasticity of outer retinal circuitry following genetic removal of horizontal cells. *The Journal of Neuroscience*, 33(45), 17847-17862.

- Keller, A.L., Zeidler, D., and Kemen, T. (2014). "High throughput data acquisition with a multi-beam SEM," in *SPIE Scanning Microscopies*, pp. 92360B1-92360B6.
- Kessel, R.G. (1969). Cytodifferentiation in the *Rana pipiens* oocyte. I. Association between mitochondria and nucleolus-like bodies in young oocytes. *The Journal of Ultrastructural Research*, 28, 61-77.
- Kim, J.S., Greene, M.J., Zlateski, A., Lee, K., Richardson, M., Turaga, S.C., Purcaro, M., Balkam, M., Robinson, A., Behabadi, B.F., Campos, M., Denk, W., Seung, H.S., and the EyeWriters (2014). Space-time wiring specificity supports direction selectivity in the retina. *Nature*, 509, 331-336.
- Kim, C.E., Perez, A., Perkins, G.A., Ellisman, M.H., and Dauer, W.T. (2010). A molecular mechanism underlying the neural-specific defect in torsinA mutant mice. *Proceedings of the National Academy of Sciences U.S.A.*, 107(21), 9861-9866.
- Kind, P.C., Kelly, G.M., Fryer, H.J.L., Blakemore, C., and Hockfield, S. (1997). Phospholipase C-beta1 is present in the botrysome, an intermediate compartment-like organelle, and is regulated by visual experience in cat visual cortex. *The Journal of Neuroscience*, 17, 1471-1480.
- Kirillov, O.I., and Kurilenko, L.A. (1977). Adrenal cortex of mice: circadian cycle of mitotic activity and volume of cell nuclei. *Endokrinologie*, 69(1), 112-114.
- Kirillov, O.I., and Kurilenko, L.A. (1979). Effect of ACTH on circadian periodicity of nuclear volume and mitotic division in the zona fasciculata externa of the adrenal cortex of mice. *The International Journal of Chronobiology*, 6, 51-55.
- Kishi, K. (1972). Fine structural and cytochemical observations on cytoplasmic nucleoluslike bodies in nerve cells of rat medulla oblongata. *Z. Zellforsch. Mikrosk. Anat.*, 132, 523-532.
- Kittler, J., and Illingworth, J. (1986). Minimum error thresholding. *Pattern Recognition*, 19, 41-47.
- Kleinfeld, D., Bharioke, A., Blinder, P., Bock, D.D., Briggman, K.L., Chklovskii, D.B., Denk, W., Helmstaedter, M., Kaufhold, J.P., Lee, W.-C.A., Meyer, H.S., Micheva, K.D., Oberlaender, M., Prohaska, S., Reid, R.C., Smith, S.J., Takemura, S., Tsai, P.S., and Sakmann, B. (2011). Large-scale automated histology in the pursuit of connectomes. *The Journal of Neuroscience*, 31(45), 16125-16138.
- Knott, A.B., Perkins, G., Schwarzenbacher, R., and Bossy-Wetzel, E. (2008). Mitochondrial fragmentation in neurodegeneration. *Nature Reviews Neuroscience*, 9, 505-518.
- Knott, G., Marchman, H., Wall, D., and Lich, B. (2008). Serial sectioning scanning electron microscopy of adult brain tissue using focused ion beam milling. *The Journal of Neuroscience*, 28(12), 2959-2964.

- Knott, G., and Genoud, C. (2013). Is EM dead? *The Journal of Cell Science*, 126, 4545-4552.
- Koenderink, J.J., and van Doorn, A.J. (1992). Surface shapes and curvature scales. *Image and Vision Computing*, 10(8), 557-564.
- Kremer, J.R., Mastronarde, D.N., and McIntosh, J.R. (1996). Computer visualization of three-dimensional image data using IMOD. *The Journal of Structural Biology*, 116, 71-76.
- Kreshuk, A., Straehle, C.N., Sommer, C., Koethe, U., Cantoni, M., Knott, G., and Hamprecht, F.A. (2011). Automated detection and segmentation of synaptic contacts in nearly isotropic serial electron microscopy images. *PLoS one*, 6(10).
- Kreshuk, A., Koethe, U., Pax, E., Bock, D.D., and Hamprecht, F.A. (2014). Automated detection of synapses in serial section transmission electron microscopy image stacks. *PLoS one*, 9(2), e87531.
- Kumar, R., Vazquez-Reina, A., and Pfister, H. (2010). Radon-like features and their application to connectomics. *IEEE Computer Society Workshop on Mathematical Methods in Biomedical Image Analysis 2010*, 186-193.
- Ladinsky, M., Mastronarde, D.N., McIntosh, J.R., Howell, K.E., and Staehelin, L.A. (1999). Golgi structure in three dimensions: Functional insights from the normal rat kidney cell. *The Journal of Cell Biology*, 144(6), 1135-1149.
- Lafarga, M., Berciano, M.T., Martinez-Guijarro, F.J., Mellstrom, B., Lopez-Garcia, C., and Naranjo, J.R. (1992). Fos-like expression and nuclear size in osmotically stimulated supraoptic nucleus neurons. *Neuroscience*, 50(4), 867-875.
- Lafontant, P.J., Behzad, A.R., Brown, E., Landry, P., Hu, N., and Burns, A.R. (2013). Cardiac myocyte diversity and a fibroblast network in the junctional region of the zebrafish heart revealed by transmission and serial block-face scanning electron microscopy. *PLoS one*, 8(8), e72388.
- Lamond, A.I., and Spector, D.L. (2003). Nuclear speckles: a model for nuclear organelles. *Nature Reviews Molecular Cell Biology*, 4, 605-612.
- Lammerding, J., Dahl, K.N., Discher, D.E., and Kamm, R.D. (2007). Nuclear mechanics and methods. *Methods in Cell Biology*, 83, 269-294.
- Lawrence, A., Bower, J.C., Perkins, G., and Ellisman, M.H. (2006). Transform-based backprojection for volume reconstruction of large format electron microscope tilt series. *The Journal of Structural Biology*, 154(2), 144-167.
- Leighton, S.B. (1981). SEM images of block faces, cut by a miniature microtome within the SEM – A technical note. *Scanning Electron Microscopy*, 2, 73-76.

- Le Beaux, Y.J. (1972). An ultrastructural study of a cytoplasmic filamentous body, termed Nematosome, in the neurons of the rat and cat substantia nigra. *Z. Zellforsch.*, 118, 147-155.
- Lee, D. (2013). Scientific analysis by the crowd: A system for implicit collaboration between experts, algorithms, and novices in distributed work. (Doctoral dissertation).
- Levinthal, C., and Ware, R. (1972). Three dimensional reconstruction from serial sections. *Nature*, 236, 207-210.
- Lew, G.M., Payer, A., and Quay, W.B. (1982). The pinealocyte nucleolus. Ultrastructural and stereological analysis of twenty-four-hour changes. *Cell and Tissue Research*, 224, 195-206.
- Lew, G.M., Washko, K., and Quay, W.B. (1984). Quantitation of ultrastructural twenty-four hour changes in pineal nuclear dimensions. *The Journal of Pineal Research*, 1, 61-68.
- Lewczuk, B., Nowicki, M., Prusik, M., and Przybylska-Gornowicz, B. (2004). Diurnal rhythms of pinealocyte ultrastructure, pineal serotonin content and plasma melatonin level in the domestic pig. *Folia Histochemica Et Cytobiologica*, 42(3), 155-164.
- Li, F.L., and Dickinson, H.G. (1986). The structure and function of nuclear invaginations characteristic of microsporogenesis in *Pinus banksiana*. *Annals of Botany*, 60(3), 321-330.
- Li, S.H., Gutekunst, C., Hersch, S.M., and Li, X. (1998a). Association of HAP1 isoforms with a unique cytoplasmic structure. *The Journal of Neurochemistry*, 71, 2178-85.
- Li, S.H., Gutekunst, C.A., Hersch, S.M., and Li, X.J. (1998b). Interactino of huntingtin-associated protein with dynactin P150Glued. *The Journal of Neuroscience*, 18, 1261-1269.
- Li, S.H., Li, H., Torre, E.R., Li, and X.J. (2000). Expression of huntingtin-associated protein1 in neuronal cells implicates a role in neuritic growth. *Molecular and Cellular Neuroscience*, 16, 168-183.
- Lich, B., Zhuge, X., Potocek, P., Boughorbel, F., and Mathisen, C. (2013). Bringing deconvolution algorithmic techniques to the electron microscope. *The Biophysical Journal*, 104, 500A.
- Lich, B.H., Boughorbel, F., Potocek, P., van den Boogaard, R., Hekking, L., Korkmaz, E., Cernohorsky, P., Hovorka, M., and Langhorst, M. (2014). 3D isotropic reconstruction of biological samples through cycles of physical and virtual sectioning in electron microscopy. Program No. 98.06. 2014 Neuroscience Meeting Planner. Washington, D.C.: *Society for Neuroscience*, 2014. Online.
- Lichtman, J.W., and Denk, W. (2011). The big and the small: challenges of imaging the brain's circuits. *Science*, 334, 618-623.

- Lipke, E., Hörnschemeyer, T., Pakzad, A., Booth, C.R., and Michalik, P. (2014). Serial block-face imaging and its potential for reconstructing diminutive cell systems: a case study from arthropods. *Microscopy and Microanalysis*, 20(3), 946-955.
- Liu, H.K. (1977). Two- and three-dimensional boundary detection. *Computer Graphics and Image Processing*, 6, 123-134.
- Liu, T., Jones, C., Seyedhosseini, M., and Tasdizen, T. (2013). A modular hierarchical approach to 3D electron microscopy image segmentation. *The Journal of Neuroscience Methods*, 226, 88-102.
- Lorensen, W.E., and Cline, H.E. (1987). Marching cubes: A high resolution 3D surface construction algorithm. *Computer Graphics*, 21(4), 163-169.
- Lu, W., Bushong, E.A., Shih, T.P., Ellisman, M.H., and Nicoll, R.A. (2014). The cell-autonomous role of excitatory synaptic transmission in the regulation of neuronal structure and function. *Neuron*, 78(3), 433-439.
- Lucchi, A., Smith, K., Achanta, R., Knott, G., and Fua, P. (2012). Supervoxel-based segmentation of mitochondria in EM image stacks with learned shape features. *IEEE Transactions on Medical Imaging*, 31(2), 474-486.
- Ludtke, S.H., Chen, D.-H., Song, J.-L., Chuang, D.T., and Chiu, W. (2004). Seeing GroEL at 6Å resolution by single particle electron cryomicroscopy. *Structure*, 12(7), 1129-1136.
- Luhmann, U.F.O., Lange, C.A., Robbie, S., Munro, P.M.G., Cowing, J.A., Armer, H.E.J., Luong, V., Carvalho, L.S., MacLaren, R.E., Fitzke, F.W., Bainbridge, J.W.B., and Ali, R.R. (2012). Differential modulation of retinal degeneration by *Ccl2* and *Cx3cr1* chemokine signaling. *PLoS one*, 7(4), e35551.
- Macagno, E.R., Levinthal, C., and Sobel, I. (1979). Three-dimensional computer reconstruction of neurons and neuronal assemblies. *Annual Reviews of Biophysics and Bioengineering*, 8, 323-351.
- Macke, J., Maack, N., Gupta, R., Denk, W., Schölkopf, B., and Borst, A. (2008). Contour-propagation algorithms for semi-automated reconstruction of neural processes. *The Journal of Neuroscience Methods*, 167, 349-357.
- Madeira, M.D., Sousa, N., Santer, R.M., Paula-Barbosa, M.M., and Gundersen, H.J.G. (1995). Age and sex do not affect the volume, cell numbers, or cell size of the suprachiasmatic nucleus of the rat: An unbiased stereological study. *The Journal of Comparative Neurology*, 361, 585-601.
- Malhas, A., Goulbourne, C., and Vaux, D.J. (2011). The nucleoplasmic reticulum: form and function. *Trends in Cell Biology*, 21(6), 362-373.
- Malhas, A., and Vaux, D.J. (2014). "Nuclear envelope invaginations and Cancer," in *Cancer Biology and the Nuclear Envelope. Recent Advances May Elucidate Past Paradoxes*. Springer, New York. pp. 523-535.

Marabini, R., and Carazo, J.-M. (1994). Pattern recognition and classification of images of biological macromolecules using artificial neural networks. *The Biophysical Journal*, 66, 1804-1814.

Martell, J.D., Deerinck, T.J., Sancak, Y., Poulos, T.L., Mootha, V.K., Sosinsky, G.E., Ellisman, M.H., and Ting, A.Y. (2012). Engineered ascorbate peroxidase as a genetically encoded reporter for electron microscopy. *Nature Biotechnology*, 30, 1143-1148.

Marx, V. (2013). Neurobiology: brain mapping in high resolution. *Nature*, 503, 147-152.

Masri, S., Cervantes, M., and Sassone-Corsi, P. (2013). The circadian clock and cell cycle: Interconnected biological circuits. *Current Opinion in Cell Biology*, 25, 730-734.

Mastronarde, D.N. (1997). Dual-axis tomography: An approach with alignment methods that preserve resolution. *The Journal of Structural Biology*, 120, 343-352.

Mastronarde, D.N. (2003). SerialEM: A program for automated tilt series acquisition on Tecnai microscopes using prediction of specimen position. *Miroscopy and Microanalysis*, 9(Suppl. 2), 1182-1183.

Matthews, B.W. (1975). Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2), 442-451.

Mauger, J.-P. (2012). Role of the nuclear envelope in calcium signaling. *Biology of the Cell*, 104, 70-83.

Mayhew, T.M., and Astle, D. (1997). Photoreceptor number and outer segment disk membrane surface area in the retina of the rat: stereological data for whole organ and average photoreceptor cell. *The Journal of Neurocytology*, 26, 53-61.

McComb, T., Cairncross, O., Noske, A.B., Wood, D.L.A., Marsh, B.J., and Ragan, M.A. (2009). Illoura: a software tool for analysis, visualization and semantic querying of cellular and other spatial biological data. *Bioinformatics*, 25(9), 1208-1210.

McEwen, B.F., and Marko, M. (2001). The emergence of electron tomography as an important tool for investigating cellular ultrastructure. *The Journal of Histochemistry and Cytochemistry*, 49(5), 553-563.

Messaoudil, C., Boudier, T., Sorzano, C.O.S., and Marco, S. (2012). TomoJ: Tomography software for three-dimensional reconstruction in transmission electron microscopy. *BMC Bioinformatics*, 8, 288.

Metzger, S., Rong, J., Nguyen, H.P., Cape, A., Tomiuk, J., Soehn, A.S., Propping, P., Freudenberg-Hua, Y., Freudenberg, J., Tong, L., Li, S.H., Li, X.J., and Riess, O. (2008). Huntingtin-associated protein-1 is a modifier of the age-at-onset of Huntington's disease. *Human Molecular Genetics*, 17, 1137-1146.

- Michevam K.D., and Smith, S.J. (2007). Array tomography: a new tool for imaging the molecular architecture and ultrastructure of neural circuits. *Neuron*, 55, 25-36.
- Miller, B.H., McDearmon, E.L., Panda, S., Hayes, K.R., Zhang, J., Andrews, J.L., Antoch, M.P., Walker, J.R., Esser, K.A., Hogenesch, J.B., and Takahashi, J.S. (2007). Circadian and CLOCK-controlled regulation of the mouse transcriptome and cell proliferation. *Proceedings of the National Academy Of Sciences U.S.A.*, 104(9), 3342-3347.
- Mishchenko, Y., Hu, T., Spacek, J., Mendenhall, J., Harris, K.M., and Chklovskii, D.B. (2010). Ultrastructural analysis of hippocampal neuropil from the connectomics perspective. *Neuron*, 67, 1009-1020.
- Miyawaki, A., Llopis, J., Heim, R., McCaffery, J.M., Adams, J.A., Ikura, M., and Tsien, R.Y. (1997). Fluorescent indicators for Ca<sup>2+</sup> based on green fluorescent proteins and calmodulin. *Nature*, 388, 882-887.
- Moens, P.B., and Moens, T. (1981). Computer measurements and graphics of three-dimensional cellular ultrastructure. *The Journal of Ultrastructure Research*, 75, 131-141.
- Mohammadi-Gheidari, and Kruit, A. (2011). Electron optics of multi-beam scanning electron microscope. *Nuclear Instruments and Methods A*, 645, 60-67.
- Moore, H.L., Chen, J., Gibson, E., Donelan, J.M., and Beg, M.F. (2010). A semi-automated method for identifying and measuring myelinated nerve fibers in scanning electron microscope images. *The Journal of Neuroscience Methods*, 201, 149-158.
- Moore, R.Y., Speh, J.C., Leak, R.K. (2002). Suprachiasmatic nucleus organization. *Cell and Tissue Research*, 309, 89-98.
- Mori, M., Ishikawa, G., Takeshita, T., Goto, T., Robinson, J.M., and Takizawa, T. (2006). Ultrahigh-resolution immunofluorescence microscopy using ultrathin cryosections: subcellular distribution of caveolin-1alpha and CD31 in human placental endothelial cells. *The Journal of Electron Microscopy*, 55, 107-112.
- Morin, L.P. (2007). SCN organization revisited. *The Journal of Biological Rhythms*, 22, 3-13.
- Morin, L.P. (2013). Neuroanatomy of the extended circadian rhythm system. *Experimental Neurology*, 243, 4-20.
- Mohawk, J.A., Green, C.B., and Takahashi, J.B. (2012). Central and peripheral circadian clocks in mammals. *Annual Review of Neuroscience*, 35, 445-462.
- Morales, J., Alonso-Nanclares, L., Rodriguez, J.-R., DeFelipe, J., Rodriguez, A., and Merchán-Pérez, A. (2011). Espina: A tool for the automated segmentation and counting of synapses in large stacks of electron microscopy images. *Frontiers in Neuroanatomy*, 5(18).

- Motoi, Y., Aizama, T., Haga, S., Nakamura, S., Namba, Y., and Ikeda, K. (1999). Neuronal localization of a novel mosaic apolipoprotein E receptor, LR11, in rat and human brain. *Brain Research*, 833, 209-215.
- Motskin, M., Müller, K., Genoud, C., Monteith, A.G., and Skepper, J.N. (2011). The sequestration of hydroxyapatite nanoparticles by human monocyte-macrophages in a compartment that allows free diffusion with the extracellular environment. *Biomaterials*, 32(35), 9470-9482.
- Müller, O.M., and Gerber, H.B. (1985). Circadian changes of the rat pancreas acinar cell. A quantitative morphological investigation. *The Scandinavian Journal of Gastroenterology*, 20(s112), 12-19.
- Mun, J.Y., Jeong, S.Y., Kim, J.H., Han, S.S., and Kim, I.-H. (2010). A low fluence Q-switched Nd:YAG laser modifies the 3D structure of melanocyte and ultrastructure of melanosome by subcellular-selective photothermolysis. *The Journal of Electron Microscopy*, dfq068.
- Murakami, G., and Uchiyama, Y. (1986). Bimodal variations in subcellular structures of rat thyroid follicular cells during 24 hours: fine structural and morphometric studies. *The American Journal of Anatomy*, 175, 1-13.
- Murphy, G.E., Lowekamp, B.C., Zervas, P.M., Chandler, R.J., Narasimha, R., Venditti, C.P. and Subramaniam, S. (2010). Ion-abrasion scanning electron microscopy reveals distorted liver mitochondrial morphology in murine methylmalonic academia. *The Journal of Structural Biology*, 171, 125-132.
- Mustafi, D., Avishai, A., Avishai, N., Engel, A., Heuer, A., and Palczewski, K. (2011). Serial sectioning for examination of photoreceptor cell architecture by focused ion beam technology. *The Journal of Neuroscience Methods*, 198, 70-76.
- Mustafi, D., Kevany, B.M., Genoud, C., Bai, X., and Palczewski, K. (2013). Photoreceptor phagocytosis is mediated by phosphoinositide signaling. *The FASEB Journal*, 27(11), 4585-4595.
- Myronenko, A., and Song, X. (2012). Point set registration: coherent point drift. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(12), 2262-2275.
- Najm, F.J., Lager, A.M., Zaermba, A., Wyatt, K., Caprariello, A.V., Factor, D.C., Tarl, R.T., Maeda, T., Miller, R.H., and Tesar, P.J. (2013). Transcription factor-mediated reprogramming of fibroblasts to expandable, myelinogenic oligodendrocyte progenitor cells. *Nature Biotechnology*, 31, 426-433.
- Nanguneri, S., Flottmann, B., Horstmann, H., Heilemann, M., and Kuner, T. (2012). Three-dimensional, tomographic super-resolution fluorescence imaging of serially sectioned thick samples. *PLoS one*, 7(5), e38098.
- Narashima, R., Ouyang, H., Gray, A., McLaughlin, S.W., and Subramaniam, S. (2009). Automatic joint classification and segmentation of whole cell 3D images. *Pattern Recognition*, 42, 1067-1079.



- Nelson, W.G., Pienta, K.J., Barrack, E.R., and Coffey, D.S. (1986). The role of the nuclear matrix in the organization and function of DNA. *Annual Review of Biophysics and Biophysical Chemistry*, 15, 457-75.
- Nemeth, A., and Langst, G. (2011). Genome organization in and around the nucleolus. *Trends in Genetics*, 27, 149-156.
- Newman, G.R., and Hobot, J.A. (1999). Resins for combined light and electron microscopy: A half century of development. *The Histochemical Journal*, 31, 495-505.
- Nicastro, D., Schwartz, C., Pierson, J., Gaudette, R., Porter, M.E., and McIntosh, J.R. (2006). The molecular architecture of axonemes revealed by cryoelectron tomography. *Science*, 313, 944-948.
- Nickell, S., Forster, F., Linaroudis, A., Net, W.D., Beck, F., Hegerl, R., Baumesiter, W., and Plitzko, J.M. (2005). TOM software toolbox: Acquisition and analysis for electron tomography. *The Journal of Structural Biology*, 149, 227-234.
- Nguyen, J.V., Soto, I., Kim, K.-Y., Bushong, E.A., Oglesby, E., Valiente-Soriano, F.J., Yang, Z., Davis, C.O., Bedont, J.L., Son, J.L., Wei, J.O., Buchman, V.L., Zack, D.J., Vidal-Sanz, M., Ellisman, M.H., and Marsh-Armstrong, N. (2011). Myelin transition zone astrocytes are constitutively phagocytic and have synuclein dependent reactivity in glaucoma. *Proceedings of the National Academy of Science U.S.A.*, 108(3), 1176-1181.
- Nogales, E., Wolf, S.G., and Downing, K.H. (1998). Structure of the alpha beta tubulin dimer by electron crystallography. *Nature*, 391, 199-203.
- Noske, A.B., Costin, A.J., Morgan, G.P., and Marsh, B.J. (2008). Expedited approaches to whole cell electron tomography and organelle mark-up in situ in high-pressure frozen pancreatic islets. *The Journal of Structural Biology*, 161(3), 298-313.
- Oberti, D., Kirschmann, M.A., and Hahnloser, R.H.R. (2011). Projection neuron circuits resolved using correlative array tomography. *Frontiers in Neuroscience*, 5, 50.
- O'Connell, M.K., Murthy, S., Phan, S., Xu, C., Buchanan, J., Spilker, R., Dalman, R.L., Zarins, C.K., Denk, W., Taylor, C.A. (2008). The three-dimensional micro- and nanostructure of the aortic medial lamellar unit measured using 3D confocal and electron microscopy imaging. *Matrix Biology*, 27(3), 171-181.
- Ollion, J., Cochenec, J., Loll, F., Escude, C., and Boudier, T. (2013). TANGO: A generic tool for high-throughput 3D image analysis for studying nuclear organization. *Bioinformatics*, btt276.
- Otsu, N. (1979). A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1), 62-66.
- Ou, H.D., Kwiatkowski, W., Deerinck, T.J., Noske, A., Blain, K.Y., Land, H.S., Soria, C., Powers, C.J., May, A.P., Shu, X., Tsien, R.Y., Fitzpatrick, J.A.J., Long, J.A., Ellisman, M.H., Choe, S., and O'Shea, C.C. (2012). A structural basis for the assembly and functions of a viral polymer that inactivates multiple tumor suppressors. *Cell*, 151(2), 304-319.

- Padeken, J., and Heun, P. (2014). Nucleolus and nuclear periphery: Velcro for heterochromatin. *Current Opinion in Cell Biology*, 28, 54-60.
- Palade, G.E. (1952). The fine structure of mitochondria. *The Anatomical Record*, 114, 427-451.
- Palade, G.E., and Porter, K.R. (1954). Studies on the endoplasmic reticulum. I. Its identification in cells in situ. *The Journal of Experimental Medicine*, 100, 641-656.
- Palay, S.L., and Palade, G.E. (1955). The fine structure of neurons. *The Journal of Biophysical and Biochemical Cytology*, 1, 69-88.
- Palmer, C.M., and Löwe, J. (2014). A cylindrical specimen holder for electron cryotomography. *Ultramicroscopy*, 137, 20-29.
- Panda, S., Nayak, S.K., Campo, B., Walker, J.R., Hogenesch, J.B., and Jegla, T. (2005). Illumination of the melanopsin signaling pathway. *Science*, 307(5709), 600-604.
- Pannese, E. (1999). The Golgi stain: Invention, diffusion and impact on neurosciences. *The Journal of the History of the Neurosciences*, 8(2), 132-140.
- Paridaen, J.T.M.L., Wilsch-Braüninger, M., and Huttner, W.B. (2013). Asymmetric inheritance of centrosome-associated primary cilium membrane directs ciliogenesis after cell division. *Cell*, 155(2), 333-344.
- Pascual, A., Barcena, M., Merelo, J.J., Carazo, J.-M. (2000). Mapping and fuzzy classification of macromolecular images using self-organizing neural networks. *Ultramicroscopy*, 84, 85-99.
- Patterson, G., Davidson, M., Manley, S., Lippincott-Schwartz, J. (2010). Superresolution imaging using single-molecule localization. *Annual Review of Physical Chemistry*, 61, 345-367.
- Paytubi, S., Wang, X., Lam, Y.W., Izquierdo, L., Hunter, M.J., Jan, E., Hundal, H.S., and Proud, C.G. (2009). ABC50 promotes translation initiation in mammalian cells. *The Journal of Biological Chemistry*, 284, 24061-24073.
- Pébusque, M.-J., Robaglia, A., and Seïte, R. (1981a). Diurnal rhythm of nucleolar volume in sympathetic neurons of the rat superior cervical ganglion. *The European Journal of Cell Biology*, 24, 128-130.
- Pébusque, M.-J., and Seïte, R. (1981b). Evidence of a circadian rhythm in nucleolar components of rat superior cervical ganglion neurons with particular reference to the fibrillar centers: an ultrastructural and stereological analysis. *The Journal of Ultrastructure Research*, 77, 83-92.
- Pébusque, M.-J., and Seïte, R. (1985). Ultrastructure and stereological analysis of nucleoli of rat nodose ganglion neuron during a 24-h period: a comparison with sympathetic

neurons of the rat superior cervical ganglion. *The Journal of the Autonomic Nervous System*, 13, 91-98.

Peddie, C.J., Blight, K., Wilson, E., Melia, C., Marrison, J., Carzaniga, R., Domart, M.-C., O'Toole, P., Larijani, B., and Collinson, L.M. (2014). Correlative and integrated light and electron microscopy of in-resin GFP fluorescence, used to localize diacylglycerol in mammalian cells. *Ultramicroscopy*, 143, 3-14.

Peddie, C.J., and Collinson, L.M. (2014). Exploring the third dimension: Volume electron microscopy comes of age. *Micron*, 61, 9-19.

Pedlar, C., and Tilly, R. (1966). A new method of serial reconstruction from electron micrographs. *The Journal of the Royal Microscopical Society*, 86(2), 189-197.

Pellettieri, J., Fitzgerald, P., Watanabe, S., Mancuso, J., Green, D.R., and Alvarado, A.S. (2010). Cell death and tissue remodeling in planarian regeneration. *Developmental Biology*, 338(1), 76-85.

Perkins, G.A., Renken, C.W., Song, J.Y., Frey, T.G., Young, S.J., Lamont, S., Martone, M.E., Lindsey, S., and Ellisman, M.H. (1997). Electron tomography of large, multicomponent biological structures. *The Journal of Structural Biology*, 120(3), 219-227.

Perkins, G.A., Sun, M.G., and Frey, T.G. (2009). Chapter 2 correlated light and electron microscopy/electron tomography of mitochondria *in situ*. *Methods in Enzymology*, 456, 29-52.

Pettersen, E.F., Goddard, T.D., Huang, C.C., Couch, G.S., Greenblatt, D.M., Meng, E.C., and Ferrin, T.E. (2004). UCSF Chimera – A visualization system for exploratory research and analysis. *The Journal of Computational Chemistry*, 25(13), 1605-1612.

Pfeifer, C.R., Shomorony, A., Aronoba, M.A., Zhang, G., Cai, T., Hu, X., Notkins, A.L., and Leapman, R.D. (2014). Quantitative analysis of mouse pancreatic islet architecture by serial block-face SEM. *The Journal of Structural Biology*,

Phan, S., Lawrence, A., Molina, T., Lanman, J., Berlanga, M., Terada, M., Kulungowski, A., Obayashi, J., and Ellisman, M.H. (2012). TxBR montage reconstruction for large field electron tomography. *The Journal of Structural Biology*, 180(1), 154-164.

Philippens, K.M.H. (1980). "Synchronization of rhythms to meal timing," in *Chronobiology: Principles and Applications to Shifts in Schedules*, pp. 403-416. Sijthoff and Noordhoff, Alphen aan den Rijn, The Netherlands.

Pielage, J., Cheng, L., Fetter, R.D., Carlton, P.M., Sedat, J.W., and Davis, G.W. (2008). A presynaptic giant ankyrin stabilizes the NMJ through regulation of presynaptic microtubules and transsynaptic cell adhesion. *Neuron*, 58, 195-209.

Pinali, C., Bennett, H., Davenport, J.B., Trafford, A.W., and Kitmitto, A. (2013). 3-D reconstruction of the cardiac sarcoplasmic reticulum reveals a continuous network linking

T-tubules: this organization is perturbed in heart failure. *Circulation Research*, 6(6), 845-858.

Pinheiro, I., Margueron, R., Shukeir, N., Eisold, M., Fritzsche, C., Richter, F.M., Mittler, G., Genoud, C., Goyama, S., Kurokawa, M., Son, J., Reinberg, D., Lachner, M., and Januwein, T. (2012). Prd3m and Prdm16 are H3K9me1 methyltransferases required for mammalian heterochromatin integrity. *Cell*, 150(5), 948-960.

Pingel, J., Lu, Y., Starborg, T., Fredberg, U., Langberg, H., Nedergaard, A., Weis, M., Eyre, D., Kjaer, M., and Kadler, K.E. (2014). 3-D ultrastructure and collagen composition of healthy and overloaded human tendon: evidence of tenocyte and matrix buckling. *The Journal of Anatomy*, 224(5), 548-555.

Pizer, S.M., Amburn, E.P., Austin, J.D., Cromartie, R., Geselowitz, A., Greer, T., ter Haar Romeny, B.M., Zimmerman, J.B., and Zuiderveld, K. (1987). Adaptive histogram equalization and its variations. *Computer Vision, Graphics and Image Processing*, 39, 355-368.

Plaza, S.M., Parag, T., Huang, G.B., Olbris, D.J., Saunders, M.A., and Rivlin, P.K. (2014). Annotating synapses in large EM datasets. *arXiv preprint*, 1409.1801.

Plaza, S.M., Scheffer, L.K., and Chklovskii, D.B. (2014). Toward large-scale connectome reconstructions. *Current Opinion in Neurobiology*, 25, 201-210.

Plaza, S.M., Scheffer, L.K., and Saunders, M. (2012). Minimizing manual image segmentation turn-around time for neuronal reconstruction by embracing uncertainty. *PLoS one*, 7(9), e44448.

Pollier, J., Moses, T., Gonzalez-Guzman, M., De Geyter, N., Lippens, S., Vanden Bossche, R., Marhavy, P., Kremer, A., Morreel, K., Guerin, C.J., Tava, A., Oleszek, W., Thevelein, J.M., Campos, N., Goormachtig, S., and Goossens, A. (2013). The protein quality control system manages plant defence compound synthesis. *Nature*, 504(7478), 148-152.

Porter, K.R., and Blum, J. (1953). A study in microtomy in electron microscopy. *The Anatomical Record*, 117, 685-710.

Porter, K.R., Claude, A., and Fullam, E.F. (1945). A study of tissue culture cells by electron microscopy: Method and preliminary observations. *The Journal of Experimental Medicine*, 81(3), 233-246.

Poulet, A., Arganda-Carreras, I., Legland, D., Probst, A.V., Andrey, P., and Tatout, C. (2014). NucleusJ: an ImageJ plugin for quantifying 3D images of interphase nuclei. *Bioinformatics*, btu774.

Powers, D.M. (2011). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness & correlation. *The Journal of Machine Learning Technologies*, 2(1), 37-63.

- Powner, M.B., Scott, A., Zhu, M., Munro, P.M.G., Foss, A.J.E., Hageman, G.S., Gillies, M.C., and Fruttiger, M. (2011). Basement membrane changes in capillaries of the ageing human retina. *The British Journal of Ophthalmology*, bjo-2011.
- Price, D.L., Chow, S.K., MacLean, N.A.B., Hakozaiki, H., Peltier, S., Martone, M.E., and Ellisman, M.H. (2006). High-resolution large-scale mosaic imaging using multiphoton microscopy to characterize transgenic mouse models of human neurological disorders. *Neuroinformatics*, 4(1), 65-80.
- Prothero, J., and Prothero, J. (1982). Three-dimensional reconstruction from serial sections. I. A portable microcomputer-based software package in Fortran. *Computers and Biomedical Research*, 15, 598-604.
- Rezakhaniha, R., Fonck, E., Genoud, C., and Stergiopoulos, N. (2011). Role of elastin anisotropy in structural strain energy functions of arterial tissue. *Biomechanics and Modeling in Mechanobiology*, 10, 599-611.
- Rieger, B., Timmermans, F.J., van Vliet, L.J., and Verbeek, P.W. (2004). On curvature estimation of ISO surfaces in 3D gray-value images and the computation of shape descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(8), 1088-1094.
- Robaglia, A., and Seite, R. (1985). Changes in nucleoli and nucleolar fibrillar centres of chromaffin cells in rat adrenal medulla over a 24-hour period: an ultrastructural and stereological analysis. *The Journal of Cell Science*, 77, 255-262.
- Romero, E., Cuisenaire, O., Deneff, J.F., Delbeke, J., Macq, B., and Veraart, C. (2000). Automatic morphometry of nerve histological sections. *The Journal of Neuroscience Methods*, 97, 111-122.
- Rouquette, J., Genoud, C., Vazquez-Nin, G.H., Kraus, B., Cremer, T., and Fakan, S. (2009). Revealing the high-resolution three-dimensional network of chromatin and interchromatin space: A novel electron-microscopic approach to reconstructing nuclear architecture. *Chromosome Research*, 17(6), 801-810.
- Röver, S., and Philippens, K.M.H. (1979). Circadian rhythm of binuclear rat liver cells. Response to phase-shifted light-dark cycle. *Chronobiologia*, 6, 149.
- Rust, M.J., Bates, M., Zhuang, X. (2006). Sub-diffraction-limit imaging by stochastic optical reconstruction microscopy (STORM). *Nature Methods*, 3, 793-95.
- Seung, H.-S. (2013). *Connectome: How the Brain's Wiring Makes Us Who We Are*. Houghton Mifflin Harcourt. New York, NY, U.S.A.
- Seyedhosseini, M., Ellisman, M.H., and Tasdizen, T. (2013a). "Segmentation of mitochondria in electron microscopy images using algebraic curves," in *2013 IEEE 10<sup>th</sup> International Symposium on Biomedical Imaging: From Nano to Macro*. San Francisco, CA, U.S.A.

- Seyedhosseini, M., Sajjadi, M., and Tasdizen, T. (2013b). Image segmentation with cascaded hierarchical models and logistic disjunctive normal networks. *2013 IEEE International Conference on Computer Vision*, 2168-2175.
- Schalek, R., Kasthuri, N., Hayworth, K., Berger, D., Tapia, J.C., Morgan, J.L., Turaga, S.C., Fagerholm, E., Seung, H.S., and Lichtman, J.W. (2011). Development of high-throughput, high-resolution 3D reconstruction of large-volume biological tissue using automated tape collection ultramicrotomy and scanning electron microscopy. *Microscopy and Microanalysis*, 17(Suppl 2), 966-967.
- Schindelin, J., Agranda-Carreras, I., Frise, E., Kaynig, V., Longair, M., Pietzsch, T., Preibisch, S., Rueden, C., Saalfeld, S., Schmid, B., Tinevez, J.-Y., White, D.J., Hartenstein, V., Eliceiri, K., Tomancak, P., and Cardona, A. (2012). Fiji: An open-source platform for biological-image analysis. *Nature Methods*, 9, 676-682.
- Schwartz, C.L., Heumann, J.M., Dawson, S.C., and Hoenger, A. (2012). A detailed, hierarchical study of *Giardia lamblia*'s ventral disc reveals novel microtubule-associated protein complexes. *PLoS one*, 7(8), e43783.
- Senft, S.L. (2011). A brief history of neuronal reconstruction. *Neuroinformatics*, 9, 119-128.
- Shinoda, K., Mori, S., Ohtsuki, T., and Osawa, Y. (1992). An aromatase-associated cytoplasmic inclusion, the "Stigmoid Body," in the rat brain: I. Distribution in the forebrain. *The Journal of Comparative Neurology*, 322, 360-76.
- Shinoda, K., Nagano, M., and Osawa, Y. (1993). An aromatase-associated cytoplasmic inclusion, the "Stigmoid Body," in the rat brain: II. Ultrastructure (with a review of its history and nomenclature). *The Journal of Comparative Neurology*, 329, 1-19.
- Shu, X., Lev-Ram, V., Deerinck, T.J., Qi, Y., Ramko, E.B., Davidson, M.W., Jin, Y., Ellisman, M.H., Tsien, R.Y. (2011). A genetically encoded tag for correlated light and electron microscopy of intact cells, tissues, and organisms. *PLoS Biology*, 9(4), e1001041.
- Silva, G.A. (2006). Neuroscience nanotechnology: Progress, opportunities and challenges. *Nature Reviews Neuroscience*, 7, 65-74.
- Sjöstrand, F.S. (1958). Ultrastructure of retinal rod synapses of the guinea pig eye as revealed by three-dimensional reconstructions from serial sections. *The Journal of Ultrastructure Research*, 2, 122-170.
- Sjöstrand, F.S. (1974). A search for the circuitry of directional selectivity and neural adaptation through three-dimensional analysis of the outer plexiform layer of the rabbit retina. *The Journal of Ultrastructural Research*, 49, 60-156.
- Slepchenko, B.M., Schaff, J.C., Macara, I., and Loew, L.M. (2003). Quantitative cell biology with the Virtual Cell. *TRENDS in Cell Biology*, 13(11), 570-576.
- Smith, P.J., Blunt, N., Wiltshire, M., Hoy, T., Teesdale-Spittle, P., Craven, M.R., Watson, J.V., Amos, W.B., Errington, R.J., and Patterson, L.H. (2000). Characteristics of a novel

deep red/infrared fluorescent cell-permeant DNA probe, DRAQ5, in intact human cells analyzed by flow cytometry, confocal, and multiphoton microscopy. *Cytometry*, 40(4), 280-291.

Smith, K., Carleton, A., and Lepetit, V. (2009). Fast ray features for learning irregular shapes. *IEEE 12th International Conference on Computer Vision*, 397-404.

Sommer, C., Straehle, C., Kothe, U., and Hamprecht, F.A. (2011). "ilastik: Interactive learning and segmentation toolkit," in *2011 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, 230-233.

Sorzano, C.O.S., Marabini, R., Velazquez-Muriel, J., Bilbao-Castro, J.R., Scheres, S.H.w., Carazo, J.M., and Pascual-Montano, A. (2004). XMIPP: a new generation of an open-source image processing package for electron microscopy. *The Journal of Structural Biology*, 148(2), 194-204.

Sosinsky, G.E., Gaietta, G.M., Hand, G., Deerinck, T.J., Han, A., Mackey, M., Adams, S.R., Bouwer, J., Tsien, R.Y., and Ellisman, M.H. (2003). Tetracysteine genetic tags complexed with biarsenical ligands as a tool for investigating gap junction structure and dynamics. *Cell Communication and Adhesion*, 10, 181-186.

Sosinsky, G.E., Deerinck, T.J., Greco, R., Buitenhuis, C.H., Bartol, T.M., and Ellisman, M.H. (2005). Development of a model for microphysiological simulations: Small nodes of Ranvier from peripheral nerves of mice reconstructed by electron tomography. *Neuroinformatics*, 3, 133-162.

Sosinsky, G.E., Crum, J., Jones, Y.Z., Lanman, J., Smarr, B., Terada, M., Martone, M.E., Deerinck, T.J., Johnson, J.E., and Ellisman, M.H. (2008). The combination of chemical fixation procedures with high pressure freezing and freeze substitution preserves highly labile tissue ultrastructure for electron tomography applications. *The Journal of Structural Biology*, 161(3), 359-371.

Sotelo, J.R., Garcia, R.B., and Wettstein, R. (1973). Serial sectioning study of some meiotic stages in *Scaptericus borrelli* (Grylloidea). *Chromosoma*, 42, 307-333.

Soto, G.E., Young, S.J., Martone, M.E., Deerinck, T.J., Lamont, S., Carragher, B.O., Hama, K., and Ellisman, M.H. (1994). Serial section electron tomography: A method for three-dimensional reconstruction of large structures. *NeuroImage*, 1(3), 230-243.

Staffler, B., van der Smagt, P., and Helmstaedter, M. (2014). Automated synapse detection in large-scale serial block-face electron microscopy data. Program No. 98.07. 2014 Neuroscience Meeting Planner. Washington, D.C.: *Society for Neuroscience*, 2014. Online.

Stephan, F.K., and Zucker, I. (1979). Circadian rhythms in drinking behavior and locomotor activity of rats are eliminated by hypothalamic lesions. *Proceedings of the National Academy of Sciences U.S.A.*, 69(6), 1583-1586.

- Stevens, B.J., and White, J.G. (1979). Computer reconstruction of mitochondria from yeast. *Methods in Enzymology*, 56, 718-728.
- Storch, K.F., Lipan, O., Leykin, I., Viswanathan, N., Davis, F.C., Wong, W.H., and Weitz, C.J. (2002). Extensive and divergent circadian gene expression in liver and heart. *Nature*, 417, 78-83.
- Straehle, C.N., Kothe, U., Knott, G., and Hamprecht, F.A. (2011). Carving: scalable interactive segmentation of neural volume electron microscopy images. *Medical Image Computing and Computer-assisted Intervention*, 14, 653-660.
- Su, B., Wang, X., Zheng, L., Perry, G., Smith, M.A., and Zhu, X. (2010). Abnormal mitochondrial dynamics and neurodegenerative diseases. *Biochimica et Biophysica Acta (BBA) – Molecular Basis of Disease*, 1802(1), 135-142.
- Takaoka, A., Yoshida, K., Mori, H., Hayashi, S., Young, S.J., and Ellisman, M.H. (2000). International telemicroscopy with a 3 MV ultrahigh voltage electron microscope. *Ultramicroscopy*, 83(1), 93-101.
- Takahashi, J.S., Hong, H.-K., Ko, C.H., and McDearmon, D.L. (2008). The genetics of mammalian circadian order and disorder: implications for physiology and disease. *Nature Reviews Genetics*, 9, 764-775.
- Takeuchi, I.K., and Takeuchi, Y.K. (1982). Ultrastructural and cytochemical studies on nucleolus-like bodies in early postimplantation rat embryos. *Cell and Tissue Research*, 226, 257-266.
- Tanaba, L.M., Kim, C.E., Alagem, N., and Dauer, W.T. (2009). Primary dystonia: molecules and mechanisms. *Nature Reviews Neurology*, 5, 598-609.
- Tanaka, Y., Kanai, Y., Okada, Y., Nonaka, S., Takeda, S., Harada, A., and Hirokawa, N. (1998). Targeted disruption of mouse conventional kinesin heavy chain *kif5B*, results in abnormal perinuclear clustering of mitochondria. *Cell*, 93(7), 1147-1158.
- Tang, G., Peng, L., Baldwin, P.R., Mann, D.S., Jiang, W., Rees, I., Ludtke, S.J. (2007). EMAN2: An extensible image processing suite for electron microscopy. *The Journal of Structural Biology*, 157(1), 38-46.
- Tasdizen, T., Whitaker, R., Marc, R., and Jones, B. (2005). "Enhancement of cell boundaries in transmission electron microscopy images," in *2005 IEEE International Conference on Image Processing*, 2, 129-32.
- Tasdizen, T., Seyedhosseini, M., Liu, T., Jones, C., and Jurrus, E. (2014). "Image segmentation for connectomics using machine learning," in *Computational Intelligence in Biomedical Imaging*, pp. 237-278. Springer New York.
- Tek, F.B., Kroeger, T., Mikula, S., and Hamprecht, F.A. (2014). "Automated cell nucleus detection for large-volume electron microscopy of neural tissue," in *2014 IEEE 11<sup>th</sup> International Symposium on Biomedical Imaging*, 69-72. Beijing, China.



- Thaunat, O., Granja, A.G., Barral, P., Filby, A., Montaner, B., Collinson, L., Martinez-Martin, N., Harwood, N.E., Bruckbauer, A., and Batista, F.D. (2012). Asymmetric segregation of polarized antigen on B cell division shapes presentation capacity. *Science*, 335, 475-479.
- Titze, B., and Denk, W. (2013). Automated in-chamber specimen coating for serial block-face electron microscopy. *The Journal of Microscopy*, 250(2), 101-110.
- Torre, E.R., Coleman, S., Yi, H., and Gutekunst, C. (2003). A protocol for isolation and biochemical characterization of stigmoid bodies from rat brain. *The Journal of Neuroscience Methods*, 125, 27-32.
- Tsien, R.Y. (1998). The green fluorescent protein. *Annual Review of Biochemistry*, 67, 509-544.
- Tzur, Y.B., Wilson, K.L., and Gruenbaum, Y. (2006). SUN-domain proteins: 'Velcro' that links the nucleoskeleton to the cytoskeleton. *Nature Reviews Molecular Cell Biology*, 7, 782-788.
- Valeo, T. (2014). How does the retina detect motion? More than 120,000 volunteers offer some clues. *Neurology Today*, 14(11), 50-52.
- van den Pol, T. (1980). The hypothalamic suprachiasmatic nucleus of rat: intrinsic anatomy. *The Journal of Comparative Neurology*, 191, 661-702.
- van der Horst, G.T.J., Majtjens, M., Kobayashi, K., Takano, R., Kanno, S.-I., Takao, M., de Wit, J., Verkerk, A., Eker, A.P.M., van Leenen, D., Buijs, R., Bootsma, D., Hoeijmakers, J.H.J., and Yasui, A. (1999). Mammalian Cry1 and Cry2 are essential for maintenance of circadian rhythms. *Nature*, 398, 627-630.
- van Heel, M. (1984). Multivariate statistical classification of noisy images (randomly oriented biological macromolecules). *Ultramicroscopy*, 13, 165-184.
- van Heel, M. (1989). Classification of very large electron microscopical image data sets. *Optik*, 82, 114-126.
- van Vliet, L.J., and Verbeek, P.W. (1993). Curvature and bending energy in digitized 2D and 3D images. *Proceedings of the Eighth Scandinavian Conference on Image Analysis*, 1403-1410.
- Varshney, L.R., Chen, B.L., Paniagua, E., Hall, D.H., and Chklovskii, D.B. (2011). Structural properties of the *Caenorhabditis elegans* neuronal network. *PLoS Computational Biology*, 7(2), e1001066.
- Vazquez, L., Sapiro, G., and Randall, G. (1998). "Segmenting neurons in electronic microscopy via geometric tracing," in *Proceedings of the 1998 International Conference on Image Processing*, 3, 814-818. Chicago, IL, U.S.A.

- Vihinen, H., Belevich, I., and Jokitalo, E. (2013). Three dimensional electron microscopy of cellular organelles by serial block face SEM and ET. *Microsc. Anal.*, 27, 7-10.
- Vitaladevuni, S.N., Mischenko, Y., Genkin, A., Chklovskii, D., and Harris, K. (2008). Mitochondria detection in electron microscopy images. *Workshop on Microscopic Image Analysis with Applications in Biology*, 42.
- Vitols, E., North, R.J., and Linnane, A.W. (1961). Studies on the oxidative metabolism of *Saccharomyces cerevisiae*. I. Observations on the fine structure of the yeast cell. *The Journal of Cell Biology*, 9(3), 689-699.
- Volkova, E.G., Kurchashova, S.Y., Polyakov, V.Y., and Sheval, E.V. (2011). Self-organization of cellular structures induced by the overexpression of nuclear envelope proteins: a correlative light and electron microscopy study. *The Journal of Electron Microscopy*, 60(1), 51-57.
- von Mayersbach, H. (1983). "An overview of the chronobiology of cellular morphology," in *Biological Rhythms and Medicine: Cellular, Metabolic, Physiopathologic, and Pharmacologic Aspects*, pp. 47-78. Springer New York.
- Vranceanu, F., Perkins, G.A., Terada, M., Chidavaenzi, R.L., Ellisman, M.H., and Lysakowski, A. (2012). Striated organelle, a cytoskeletal structure positioned to modulate hair-cell transduction. *Proceedings of the National Academy of Sciences U.S.A.*, 109(12), 4473-4478.
- Wacker, I., and Schroeder, R.R. (2013). Array tomography. *The Journal of Microscopy*, 252(2), 93-99.
- Wang, N., Tytell, H.D., and Ingber, D.E. (2009). Mechanotransduction at a distance: mechanically coupling the extracellular matrix with the nucleus. *Nature Reviews Molecular Cell Biology*, 10, 75-82.
- Watanabe, T., Matsuba, H., and Uchiyama, Y. (1988). Correlation of 24-hour fluctuations in renin granules of juxtaglomerular cells and in renin and angiotensinogen in blood plasma of the rat. *Cell and Tissue Research*, 254, 593-598.
- Watanabe, M., and Uchiyama, Y. (1988). Twenty-four hour variations in subcellular structures of rat pancreatic islet B-, A-, and D-cells, and of portal plasma glucose and insulin levels. *Cell and Tissue Research*, 253, 337-345.
- Weakley, B. (1969). Granular cytoplasmic bodies in oocytes of the golden hamster during the postnatal period. *Z. Zellforsch*, 101, 394-400.
- Wei, D., Jacobs, S., Modla, S., Zhang, S., Young, C.L., Cirino, R., Caplan, J., and Czymmek, K. (2012). High-resolution three-dimensional reconstruction of a whole yeast cell using focused-ion beam scanning electron microscopy. *BioTechniques*, 53, 41-48.

- Welsh, D.K., Logothetis, D.E., Meister, M., and Reppert, S.M. (1995). Individual neurons dissociated from rat suprachiasmatic nucleus express independently phased circadian firing rhythms. *Neuron*, 14, 697-706.
- Welsh, D.K., Yoo, S.-H., Liu, A.C., Takahashi, J.S., and Kay, S.A. (1994). Bioluminescence imaging of individual fibroblasts reveals persistent, independently phased circadian rhythms of clock gene expression. *Current Biology*, 14, 2289-2295.
- Welsh, D.K., Takahashi, J.K., and Kay, S.A. (2010). Suprachiasmatic nucleus: cell autonomy and network properties. *Annual Review of Physiology*, 72, 551-577.
- West, J.B., Fu, Z., Deerinck, T.J., Mackey, M.R., Obayashi, J.T., and Ellisman, M.H. (2010). Structure-function studies of blood and air capillaries in chicken lung using 3D electron microscopy. *Respiratory Physiology & Neurobiology*, 170(2), 202-209.
- White, J.G., and Amos, W.B. (1987). Confocal microscopy comes of age. *Nature*, 328, 183-184.
- White, J.G., Southgate, E., Thomson, J.N., and Brenner, S. (1986). The structure of the nervous system of the nematode *Caenorhabditis elegans*. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 314(1165), 1-340.
- Wiley, T.J., Schultz, R.L., and Gott, A.H. (1973). Computer graphics in three dimensions for perspective reconstruction of brain ultrastructure. *IEEE Transactions on Biomedical Engineering*, 4, 288-291.
- Wilke, S.A., Antonios, J.K., Bushong, E.A., Badkoobehi, A., Malek, E., Hwang, M., Terada, M., Ellisman, M.H., and Ghosh, A. (2013). Deconstructing complexity: Serial block-face electron microscopic analysis of the hippocampal mossy fiber synapse. *The Journal of Neuroscience*, 33(2), 507-522.
- Wilke, S.A., Raam, T., Antonios, J.K., Bushong, E.A., Koo, E.H., Ellisman, M.H., and Ghosh, A. (2014). Specific disruption of hippocampal mossy fiber synapses in a mouse model of familial Alzheimer's disease. *PLoS one*, 9(1), e84349.
- Wilt, B.A., Burns, L.D., Ho, E.T.W., Ghost, K.K., Mukamel, E.A., and Schnitzer, M.J. (2009). Advances in light microscopy for neuroscience. *Annual Review of Neuroscience*, 32, 435.
- Wong, J., Baddeley, D., Bushong, E.A., Yu, Z., Ellisman, M.H., Hoshijima, M., and Soeller, C. (2013). Nanoscale distribution of ryanodine receptors and caveolin-3 in mouse ventricular myocytes: dilation of T-tubules near junctions. *The Biophysical Journal*, 104(11), L22-L24.
- Worman, H.J. (2012). Nuclear lamins and laminopathies. *The Journal of Pathology*, 226(2), 316-325.

- Wright, M.L., Blanchard, L.S., Pikula, A., and Labieniec, K.E. (1995). Circadian rhythms of thyroid secretion, morphometry, and cell division in prometamorphic and climax *Rana* tadpoles. *General and Comparative Endocrinology*, 99, 75-84.
- Wu, H.-S., Barba, J., and Gil, J. (1996). An iterative algorithm for cell segmentation using short-time Fourier transform. *The Journal of Microscopy*, 184, 127-132.
- Xiao, L., Michalski, N., Kronander, E., Gjoni, E., Genoud, C., Knott, G., and Schneggenburger, R. (2013). BMP signaling specifies the development of a large and fast CNS synapse. *Nature Neuroscience*, 16, 856-864.
- Yamazaki, S., Straume, M., Tei, H., Sakaki, Y., Menaker, M., and Block, G.D. (2002). Effects of aging on central and peripheral mammalian clocks. *Proceedings of the National Academy of Sciences*, 99, 10801-10806.
- Yan, X., Sinkovits, R.S., and Baker, T.S. (2007). AUTO3DEM – An automated and high throughput program for image reconstruction of icosahedral particles. *The Journal of Structural Biology*, 157, 73-82.
- Yang, Q., Rout, M.P., and Akey, C.W. (1998). Three-dimensional architecture of the isolated yeast nuclear pore complex: Functional and evolutionary implications. *Molecular Cell*, 1(2), 223-234.
- Young, I.T., Walker, J.E., and Bowie, J.E. (1974). An analysis technique for biological shape. I. *Information and Control*, 25, 357-370.
- Young, R.D., Knupp, C., Pinali, C., Png, K.M.Y., Ralphs, J.R., Bushby, A.J., Starborg, T., Kadler, K.E., and Quantock, A.J. (2014). Three-dimensional aspects of matrix assembly by cells in the developing cornea. *Proceedings of the National Academy of Sciences U.S.A.*, 111(2), 687-692.
- Young, S.J., Royer, S.M., Groves, P.M., and Kinnamon, J.C. (1987). Three-dimensional reconstructions from serial micrographs using the IBM PC. *The Journal of Electron Microscopy Technique*, 6(2), 207-217.
- Zhang, Z. (1994). Iterative point matching for registration of free-form curves and surfaces. *The International Journal of Computer Vision*, 13(2), 119-152.
- Zhao, X., Pan, Z., Wu, J., Zhou, G., and Zeng, Y. (2010). Automatic identification and morphometry of optic nerve fibers in electron microscopy images. *Computerized Medical Imaging and Graphics*, 34(3), 179-184.
- Zhou, J., Lamichhane, S., Sterne, G., Ye, B., and Peng, H. (2013). BIOCAT: A pattern recognition platform for customizable biological image classification and annotation. *BMC Bioinformatics*, 14(291).
- Zhuravleva, E., Gut, H., Hynx, D., Marcellin, D., Bleck, C.K.E., Genoud, C., Cron, P., Keusch, J.J., Dummler, B., Esposti, M.D., and Hemmings, B.A. (2012). Acyl coenzyme A

thioesterase Them5/Acot15 is involved in cardiolipin remodeling and fatty liver development. *Molecular and Cellular Biology*, 32(14), 2685-2697.

Zink, D., Fischer, A.H., and Nickerson, J.A. (2004). Nuclear structure in cancer cells. *Nature Reviews Cancer*, 4, 677-687.

Zucker, S.W., and Hummel, R.A. (1981). A three-dimensional edge operator. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3, 324-331.

Zwerger, M., Ho, C.Y., and Lammerding, J. (2011). Nuclear mechanics in disease. *Annual Review of Biomedical Engineering*, 13, 397-428.