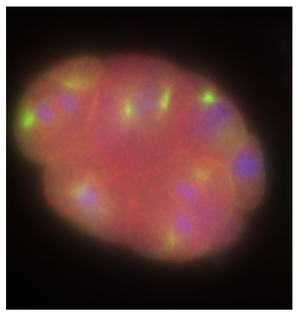
Using ImageJ to Improve Fluorescence Microscopy Images

David Adaway, Chinonso Okoli, and Dr. Chrysanthe Preza
Computational Imaging Research Laboratory
Department of Electrical and Computer Engineering, The University of Memphis

Introduction

We use microscopy to make the invisible visible – that is, to see great detail in tiny three-dimensional objects. However, the process of capturing images from a microscope is imperfect, so we use various computer programs to improve microcopy output.

Datasets



***C. elegans* embryo** – three channels (shown here in RGB form) [1]

cameraman.tif – 8-bit grayscale image [2]

Tools

ImageJ – open-source scientific image processing software [3]

Fiji – distribution of ImageJ with built-in plugins and updater [4]

- **Plugins** – additional programs that can be installed into ImageJ and provide additional functionality
- **Macros** – Java scripts that execute ImageJ functions in sequence [5]

Conclusion

We have learned about the use of computational imaging tools to recreate the microscopic imaging process and improve output images. However, the project is still a work-in-progress; we have not yet recreated the process from start to finish with our own dataset or with different microscope settings.

Acknowledgements

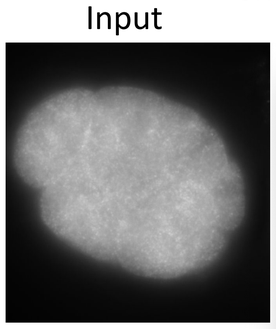
We would like to thank the other members of our Vertically Integrated Projects (VIP) team – Rosalia Nwaobi, Haleigh Sisson, and Arash Atibi – for their assistance throughout the development of our project.

Methodology & Results

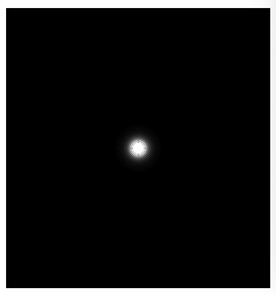
Deconvolution

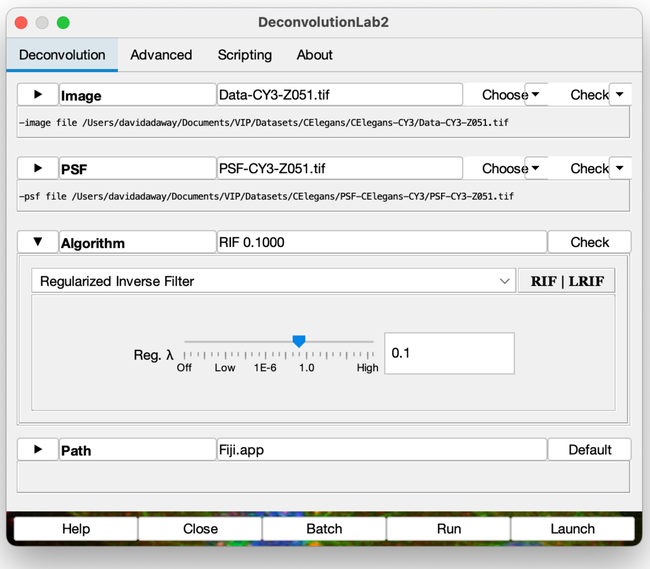
- **Convolution** – microscopy images are blurred according to a point-spread function (PSF)
- **DeconvolutionLab2** – plugin that attempts to mitigate the effects of convolution [6]

Input

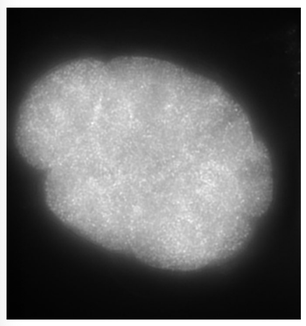


PSF





Restored output



Noising/Denoising

- **Noise** – corruption of image from microscope
- **RandomJ** – plugin that adds noise to images [7]
- **PureDenoise** and **DnCNN** – two plugins that attempt to remove noise from images [8][9]
- **CLIJ2** – plugin that calculates the mean squared error of two images [10]

Modulatory Poisson noise

PureDenoise

MSE = 34.670

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

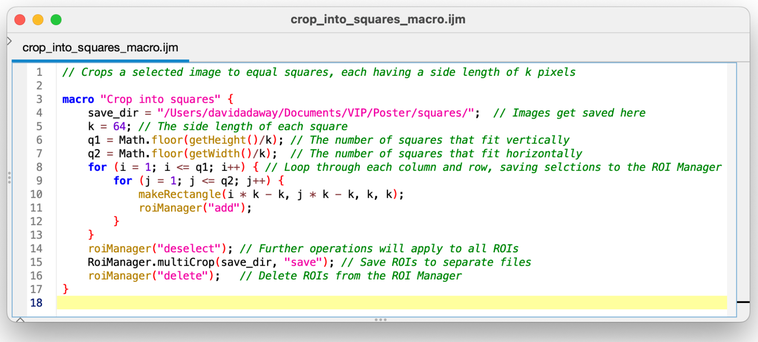
Calculation of MSE – square the difference in each pixel, sum the squares, and divide by the number of pixels

DnCNN

MSE = 73.623

Cropping/Stitching

- **Cropping** – splitting an image into smaller pieces
- **MosiacJ** – plugin that stitches cropped pieces back together [11]



MosaicJ canvas

Stitched output

References

[1] "C. elegans embryo." Accessed: Jan. 27, 2024. [Online]. Available: https://bigwww.epfl.ch/deconvolution/bio/

[2] "Image Processing Toolbox." Accessed: Jan. 27, 2024. [Online]. Available: https://www.mathworks.com/products/image.html

[3] C. A. Schneider, W. S. Rasband, and K. W. Eliceiri, "NIH Image to ImageJ: 25 years of image analysis," Nat Methods, vol. 9, no. 7, pp. 671–675, Jul. 2012, doi: 10.1038/nmeth.2089.

[4] J. Schindelin et al., "Fiji: an open-source platform for biological-image analysis," Nat Methods, vol. 9, no. 7, pp. 676–682, Jul. 2012, doi: 10.1038/nmeth.2019.

[5] J. Broeke, J. M. Mateos Pérez, and J. Pascau, Image processing with ImageJ: extract and analyze data from complex images with ImageJ, the world's leading image processing tool, Second edition. in Community experience distilled. Birmingham Mumbai: Packt Publishing open source, 2015.

[6] D. Sage et al., "DeconvolutionLab2: An open-source software for deconvolution microscopy," Methods, vol. 115, pp. 28–41, Feb. 2017, doi: 10.1016/j.ymeth.2016.12.015.

[7] "RandomJ." Accessed: Jan. 27, 2024. [Online]. Available: https://imagescience.org/meijering/software/randomj/

[8] F. Luisier, C. Vonesch, T. Blu, and M. Unser, "Fast interscale wavelet denoising of Poisson-corrupted images," Signal Processing, vol. 90, no. 2, pp. 415–427, Feb. 2010, doi: 10.1016/j.sigpro.2009.07.009.

[9] V. Mannam et al., "Real-time image denoising of mixed Poisson–Gaussian noise in fluorescence microscopy images using ImageJ," Optica, vol. 9, no. 4, p. 335, Apr. 2022, doi: 10.1364/OPTICA.448287.

[10] R. Haase et al., "CLIJ: GPU-accelerated image processing for everyone," Nat Methods, vol. 17, no. 1, pp. 5–6, Jan. 2020, doi: 10.1038/s41592-019-0650-1.

[11] P. Thévenaz and M. Unser, "User-friendly semiautomated assembly of accurate image mosaics in microscopy," Microscopy Res & Technique, vol. 70, no. 2, pp. 135–146, Feb. 2007, doi: 10.1002/jemt.20393.