*4.1 3DMembraneReconstruction software setup*

The most up-to-date version of the reconstruction software is available for free download from Zenodo (DOI: [10.5281/zenodo.1316957](https://doi.org/10.5281/zenodo.1316957)). We recommend forking the master repository and cloning it to your desktop; then you will have the freedom to make changes to the code while maintaining the ability to revert these changes back to the original format and also sync your forked repository to the master if software updates are available.

The 3DMembraneReconstruction software was written to run optimally on MATLAB versions 2015a-2016b. Ensure that one of these versions is installed and operating. Before running the software, add the 3DMembraneReconstruction folder to the MATLAB path. If using the user interface (UI), navigate to the main.m program and run it to initialize the analysis. If using the manual code (sample\_script\_membrane.m), navigate to the image file locations. Further detailed instructions on running the software are provided in the source code on GitHub.

**Appendix**

*1. Contrast adjustment compensation for lower bit-sized images*

To compensate for lower bit size, images may be adjusted to increase the range of pixel intensities represented (i.e., increase contrast). A simple MATLAB procedure to achieve this uses the function imadjust, which redistributes pixel intensity so that the extreme low and high intensity values are only 1% of the represented pixels. A sample script for image contrast adjustment is as follows:

for i = 1:size(wimg3,3)

wimg3\_adj(:,:,i)=imadjust(wimg3(:,:,i)); % adjust pixel

% intensity to increase

% overall image contrast.

end

*2. Manual boundary exclusion to prevent incorrect membrane reconstruction in dense nuclear samples*

In order to use the reconstruction software on denser nuclear samples (e.g., stem cell colonies, organoids, and whole tissue), one may need to manually exclude unwanted fluorescent boundaries. The following MATLAB script (excludeRegions.m) is a simple method for boundary exclusion that removes unwanted regions of interest throughout the z stack of processed images.

function [img\_filt] = excludeRegions(obj,regionNum)

for k = 1:regionNum % Loop for multiple regions in image to be

% excluded.

midplane = round(length(obj)/2); % Find midplane of z-stack.

mask = roipoly(obj(:,:,midplane)); % Use polygon tool to

% draw polygon around

% region to be excluded.

% When finished closing

% polygon, double click

% in center of polygon

% to save polygon mask.

if k == 1

img\_filt = obj; % Create image copy to be filtered.

end

figure

imshowpair(img\_filt(:,:,midplane),mask) % Show mask on top of

% original image to show

% the region to be

% converted to zeros.

[mask\_y, mask\_x] = find(mask); % Save mask [row, column]

% coordinates as separate vectors.

for j = 1:length(img\_filt) % For each z slice...

for i = 1:length(mask\_x) % For each mask coordinate...

img\_filt(mask\_y(i),mask\_x(i),j) = 0; % Convert the

% corresponding image

% coordinate value to

% zero.

end

end

figure

imshow(img\_filt(:,:,midplane)) % Show filtered image result.

end

end

*3. Batch processing for multiple image sets*

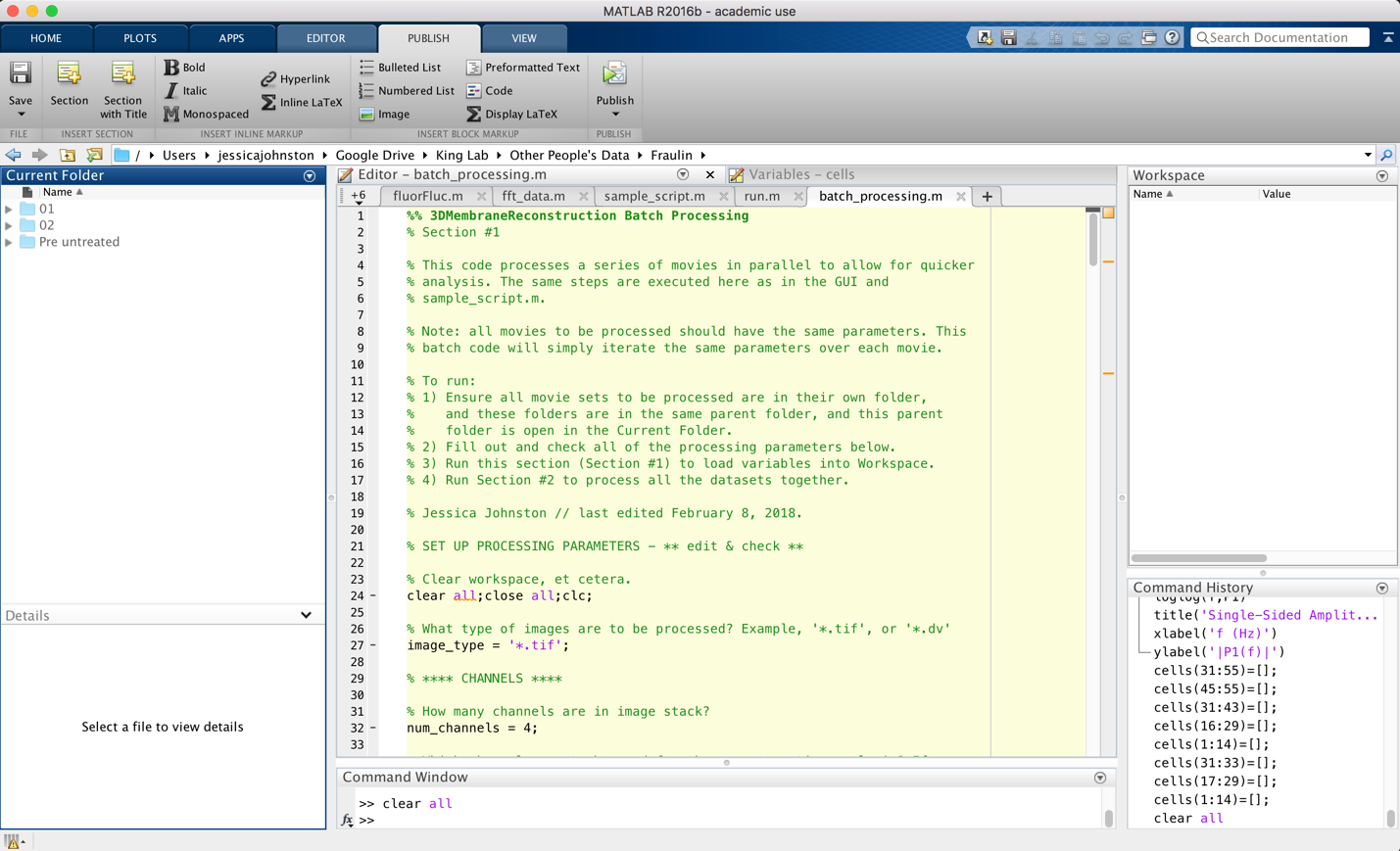
A convenient addition to the software was recently added to allow for batch processing of multiple image sets imaged using the same parameters. Batch processing allows for a series of images to be processed and analyzed in parallel. The following is a detailed description of how to run the batch processing code.

After adding the updated 3DMembraneReconstruction to your MATLAB path, open the script “batch\_processing.m”, which can be found in the 3DMembraneReconstruction parent folder. Make sure all of your movie/data files are stored as follows:

1) Each movie/file set is in its own folder.

2) Each of these folders are in one parent folder.

3) This parent folder is open in MATLAB’s Current Folder (displaying all the movie set folders):



The first section of the code includes all of the processing parameters one would like to apply to all of one’s movies/datasets. Read through each variable and make sure it is consistent with the imaging conditions (e.g., number of imaging channels, name of the fluorescent particle, etc.).

%% 3DMembraneReconstruction Batch Processing

% Section #1

% This code processes a series of movies in parallel to allow for quicker

% analysis. The same steps are executed here as in the GUI and

% sample\_script.m.

% Note: all movies to be processed should have the same parameters. This

% batch code will simply iterate the same parameters over each movie.

Follow these steps:

% To run:

% 1) Ensure all movie sets to be processed are in their own folder,

% and these folders are in the same parent folder, and this parent

% folder is open in the Current Folder.

% 2) Fill out and check all of the processing parameters below.

% 3) Run this section (Section #1) to load variables into Workspace.

% 4) Run Section #2 to process all the datasets together.

% Jessica Johnston // last edited February 8, 2018.

READ AND FILL OUT

% SET UP PROCESSING PARAMETERS - \*\* edit & check \*\*

% Clear workspace, et cetera.

clear all;close all;clc;

% What type of images are to be processed? Example, '\*.tif', or '\*.dv'

image\_type = '\*.tif';

% \*\*\*\* CHANNELS \*\*\*\*

% How many channels are in image stack?

num\_channels = 4;

% Which channels are to be used for the reconstruction analysis? If yes,

% assign to '1' for particle channel and '2' for membrane channel. If not

% using channel, assign to '0'.

channel\_1 = 0;

channel\_2 = 0;

channel\_3 = 1;

channel\_4 = 2;

% \*\*\*\* PARTICLE ANALYSIS \*\*\*\*

% For one particle analysis, set to '1'; for two particle analysis, set to

% '2'; to omit this analysis altogether, set to '0'.

particle\_channel = 1;

% Designate particle name(s).

particle\_name = 'loci';

% particle\_name2 = 'mCherry-loci';

% Designate particle size in pixels (approximate).

particle\_size = 3;

% \*\*\*\* MEMBRANE ANALYSIS \*\*\*\*

% For membrane analysis, set to '1'; to omit this, set to '0'.

membrane\_channel = 1;

% Designate membrane name.

membrane\_name = 'cut11';

% Designate membrane size in pixels (approximate).

membrane\_size = 30;

% \*\*\*\* CELL CONSTRUCTION \*\*\*

% Designate how to construct the cells - by particles only, membranes only,

% or by both particles and membranes. If yes, set to '1'; if no, set to

% '0'. Only set one variable to '1'; leave other two variable as '0'.

% Construct cells only by particles:

cell\_construct\_particle = 0;

% Construct cells only by membranes:

cell\_construct\_membrane = 0;

% Construct cells by both particles and membranes:

cell\_construct\_membrane\_particle = 1;

% How many fluorescent particles should be in each cell? Set this number as

% a means of filtering out cells with fewer/more particles.

particle\_number = 1;

% \*\*\*\* SAVING IMAGES \*\*\*\*

% Show and save each reconstruction? If so, set 'show\_and\_save' to '1'; if

% you want to run the reconstructions without showing and saving each

% individual image (much faster overall), set to '0'.

show\_and\_save = 1;

After you fill out and check each of these parameters, run this section to load the variables into the Workspace. Next run Section #2, which processes all the datasets in parallel. Do not edit anything in Section #2. All images and measurements will be saved into each data folder:

