AMaSiNe User Manual

Jun Ho Song & Woochul Choi

Correspondence: Prof. Se-Bum Paik

sbpaik@kaist.ac.kr

Manual version: 07272020

What's New

- We implemented Elastix [1,2] and multithreading functions for image registration in Step 5

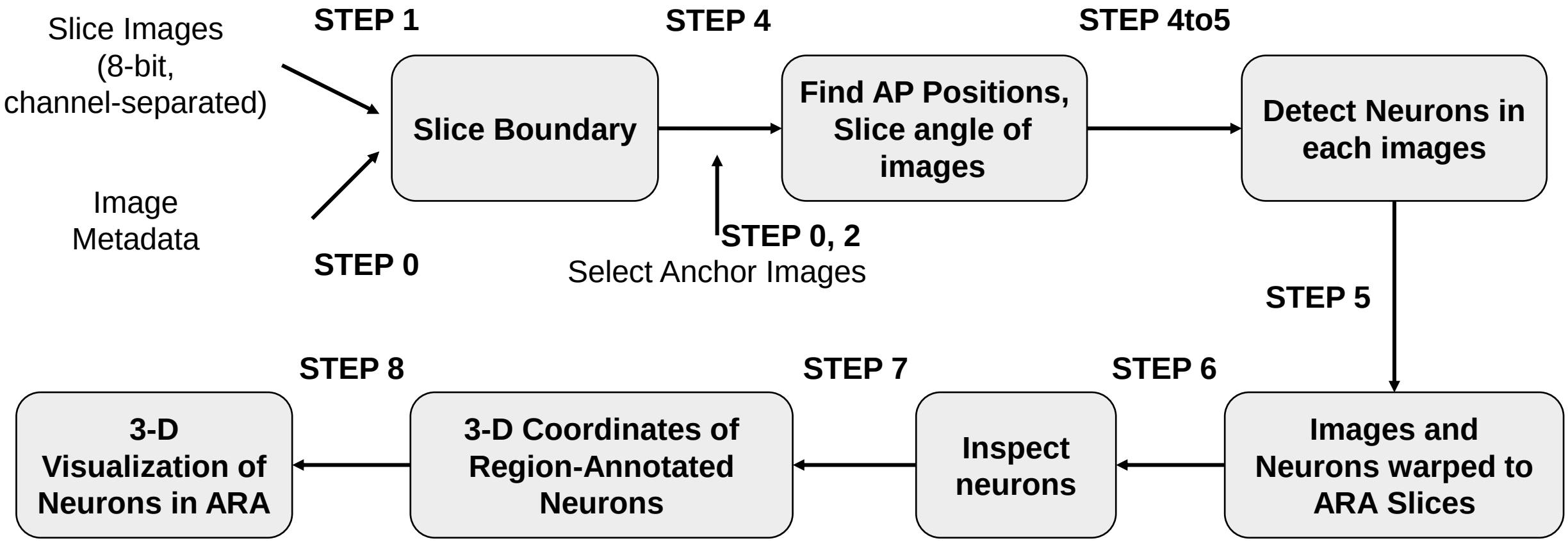
➔ > x 8 times faster

- Reported errors caused by Matlab version compatibility issues (2018,2019, and 2020) and some minor bugs were resolved

- Variables/parameters that are not used were removed from the scripts

1. Klein, Stefan, et al. "Elastix: a toolbox for intensity-based medical image registration." *IEEE transactions on medical imaging* 29.1 (2009): 196-205.
2. Rob Campbell. MATLAB Elastix (https://github.com/raacampbell/matlab_elastix), GitHub. (2020)

Algorithm pipeline of AMaSiNe



Preparation - Elastix

1. Download Elastix binary codes from:

   https://github.com/SuperElastix/elastix/releases/tag/5.0.0

   * Elastix 5.0.0 was used when developing/testing this Amasine update

2. Uncompress the file and add the unzipped folder to your system path

   Windows: Link Linux: Link

Preparation

- MATLAB (> 2017a) is required

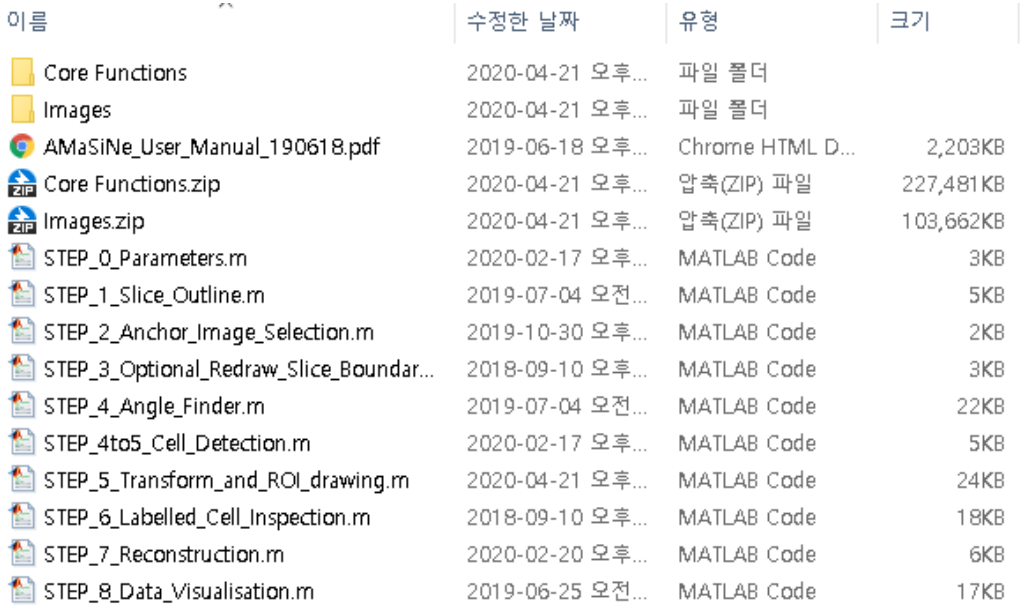
3. The following MATLAB packages are required

  - Computer Vision Toolbox
  - Image Processing Toolbox
  - Parallel Computing Toolbox

4. Make a new empty folder. Then, unzip the “**Core Functions.zip**” and “**Images.zip** (or your images)” and place the **Amasine scripts** in the same folder

MUY IMPORTANTE: Don’t put a space in your folder name

Ejemplo) “PV Cre Mouse” **X** “PV_Cre_Mouse” **O**



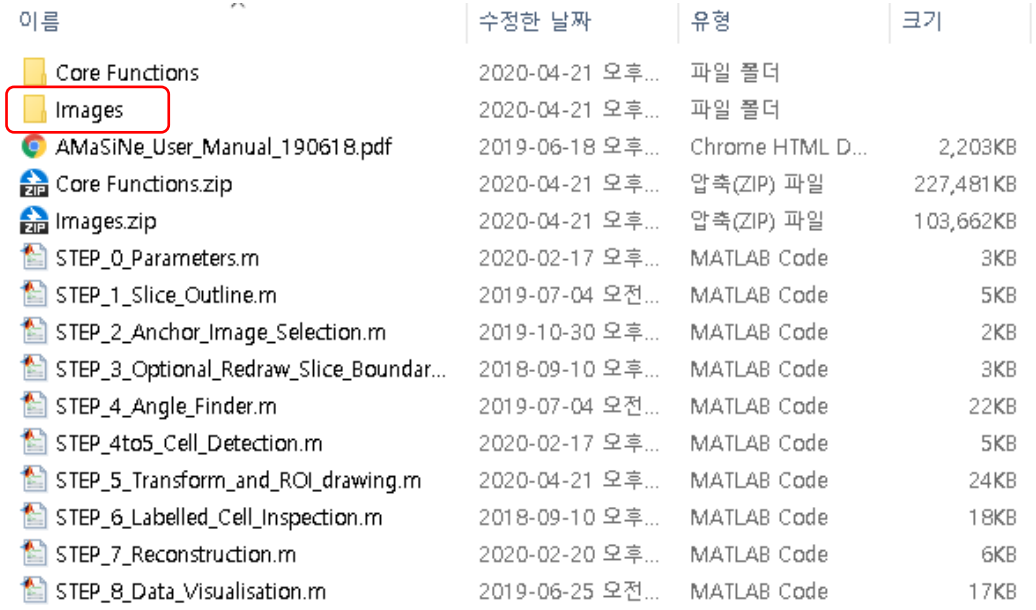
이름	수정한 날짜	유형	크기
Core Functions	2020-04-21 오후...	파일 폴더	
Images	2020-04-21 오후...	파일 폴더	
AMaSiNe_User_Manual_190618.pdf	2019-06-18 오후...	Chrome HTML D...	2,203KB
Core Functions.zip	2020-04-21 오후...	압축(ZIP) 파일	227,481KB
Images.zip	2020-04-21 오후...	압축(ZIP) 파일	103,662KB
STEP_0_Parameters.m	2020-02-17 오후...	MATLAB Code	3KB
STEP_1_Slice_Outline.m	2019-07-04 오전...	MATLAB Code	5KB
STEP_2_Anchor_Image_Selection.m	2019-10-30 오후...	MATLAB Code	2KB
STEP_3_Optional_Redraw_Slice_Boundar...	2018-09-10 오후...	MATLAB Code	3KB
STEP_4_Angle_Finder.m	2019-07-04 오전...	MATLAB Code	22KB
STEP_4to5_Cell_Detection.m	2020-02-17 오후...	MATLAB Code	5KB
STEP_5_Transform_and_ROI_drawing.m	2020-04-21 오후...	MATLAB Code	24KB
STEP_6_Labelled_Cell_Inspection.m	2018-09-10 오후...	MATLAB Code	18KB
STEP_7_Reconstruction.m	2020-02-20 오후...	MATLAB Code	6KB
STEP_8_Data_Visualisation.m	2019-06-25 오전...	MATLAB Code	17KB

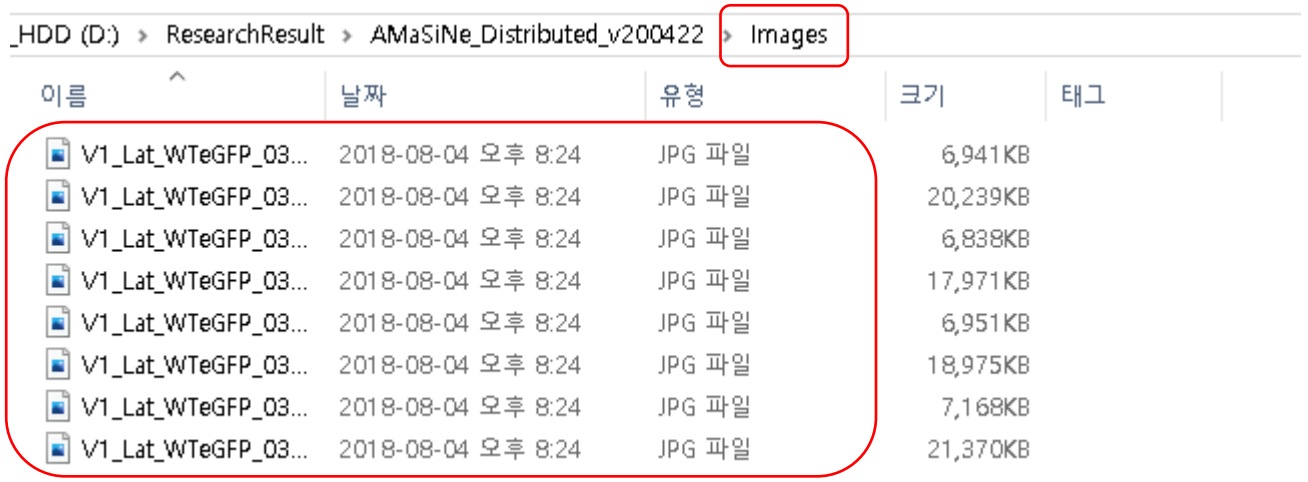
Preparation

4. In “Images”, experimental images obtained from a “single” brain should be placed

   - Make sure that other images are not included in the folder (including the mother directory)

   - If you want to analyze images from a different brain, you have to make a new folder, unzip the downloaded files, and copy&paste your images in the “Images” folder again.





5. Run MATLAB and change your directory to the mother folder

STEP_0_Parameters

- There are 16 parameters to be set before moving on;
  - Once these are set, you will need to change only 3~5 parameters to analyze image sets obtained from different mouse brains

- Double Click and open the script “STEP_0_Parameters.m”

STEP_0_Parameters

- **Image directory**

  Set “**main_folder_dir**” as the folder directory that contains all the functions and images (mother directory)

  e.g. main_folder_dir = ‘C:\Users\WChoi\Documents\MATLAB\AMaSiNe_Tutorial’

```
%% 1. Image Directory
main_folder_dir='C:₩Users₩WChoi₩Documents₩MATLAB₩AMaSiNe_Tutorial';
```

- **Image names and slice orders**

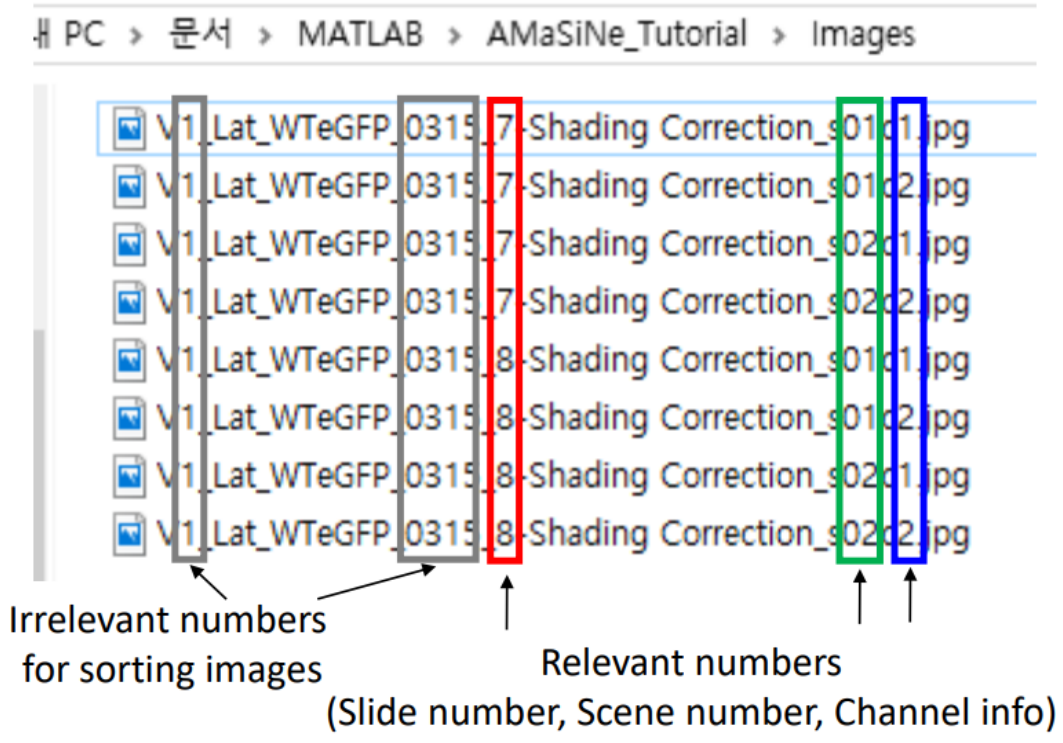
  img_format : set it as either ‘tif’ or ‘jpg’, depending on your image format
  Slice_AP_orPA : The direction in which you sliced your mouse brain
  If you sliced anterior → posterior, the value is 1
  If you sliced posterior → anterior, the value should be -1

```
%% 2. Image names and slice order
img_format ='jpg';  % if tif = 'tif' ; jpg='jpg'
Slice_AP_orPA= 1;              % If the brain is sliced from anterior to posterior, set this value = 1
                               %                          posterior to anterior, set this value = -1
```

STEP_0_Parameters

- **(Continued) Image names and slice orders**

slide_digit, scene_digit, channel_digit : determined by your 'filename' format



Let's say you obtained 4 slices of brain
s01 and s02 from 7th slide glass
s01 and s02 from 8th slide glass
and imaged with 2 channels (c1 and c2, i.e. eGFP and DAPI)

For example, you saved your image file with filename

V1_Lat_WTeGFP_0315_7-Shading Correction_s01c1

1st 2nd 3rd : Slide digit 4th : Scene digit 5th : channel digit

Then, you may define variable as

slide_digit = 3;
scene_digit = 4;
channel_digit = 5;

```
slide_digit=3;
scene_digit=4;
channel_digit= 5;
```

STEP_0_Parameters

- **Anchor Image IDs for Angle Finding**

  Leave it empty for now; you will come back to this after Step_2

- **Image Parameters**

  xy_pix = Pixel size (um/pixel)

  Name_Channels = image channel names…

  e.g. if channel 1= DAPI & channel 2= eGFP

  ➔ Name_Channels={'DAPI', 'eGFP'};

  Color_Channel_Structure = index of the channel for finding the slicing angles & slice AP pos.
  e.g. if you'd like to use DAPI for angle searching in the case of the example above,
  set this value as "1"

  Structure_stain = Staining method of the images to be used for angle searching
  Choose one of the three : 'DAPI','AutoF','Nissl'

  Color_Channel_Interest = Color channel index in which labelled cells are imaged

  e.g. in the example case mentioned above, set this value as "2" i.e. eGFP channel

STEP_0_Parameters

- **Detection Parameters**

  soma_radius = rough range of radius (not diameter) of labelled soma (um)

  cell_det_thresh = Minimum intensity difference between a cell and background for a cell to be detected
  Lower this value, you'd get a better chance of detecting cells dim
  but you also risk detecting noise as a cell

  cf) you can modify the code in "**SomaDetection0827.m**"
  to customize/implement your own algorithm for cell detection

- **Allen Reference Brain (CCFv3)**

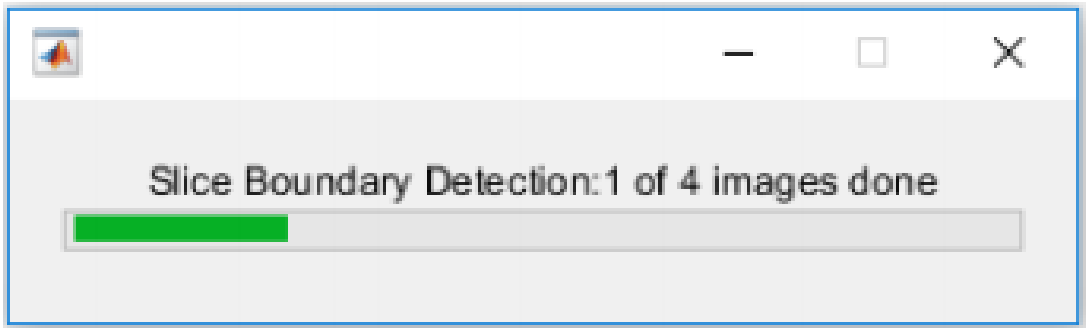
  You probably wouldn't want to change these parameters…

STEP_1_Slice_Outline

- This step is to remove image regions other than brain tissues,
  thus saving computional time and memory

- In the command window, type “STEP_1_Slice_Outline” and press Enter;
  You can also run the script in the Editor window

- A status bar showing the progress of automatic background removal will pop up



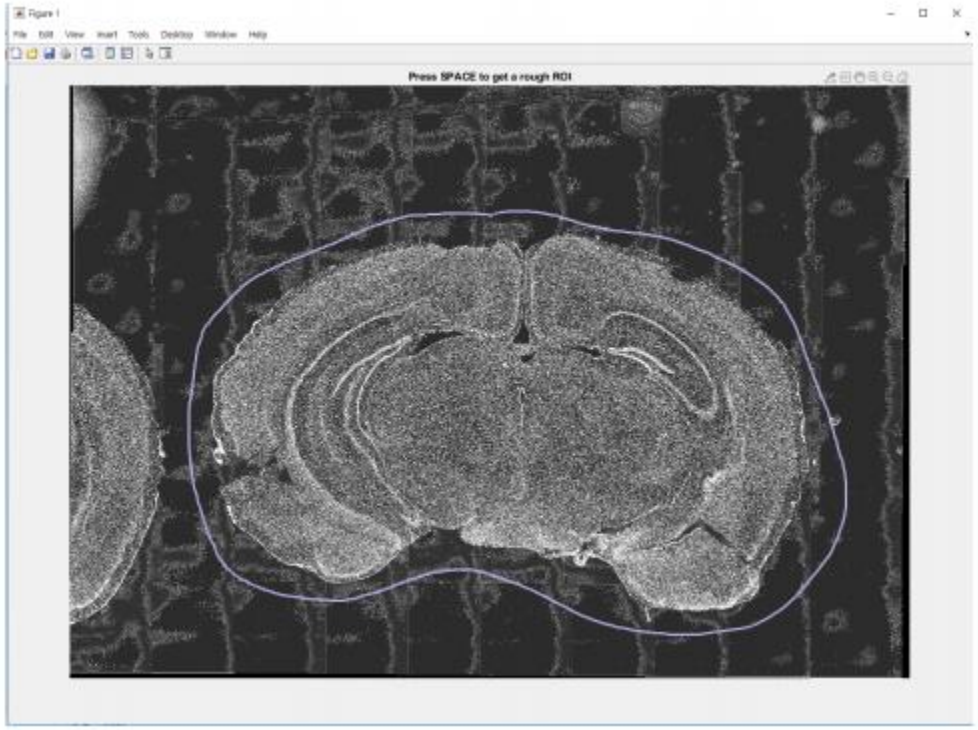
“ You have 8 images from 4 slices in the test image set”

STEP_1_Slice_Outline

- Sometimes, automatic slice boundary detection fails to give you the correct output

  ➔ For these images, you can manually draw ROUGH boundaries by clicking-and-dragging

  ➔ The algorithm will re-search for the correct slice boundaries within the rough boundaries you drew

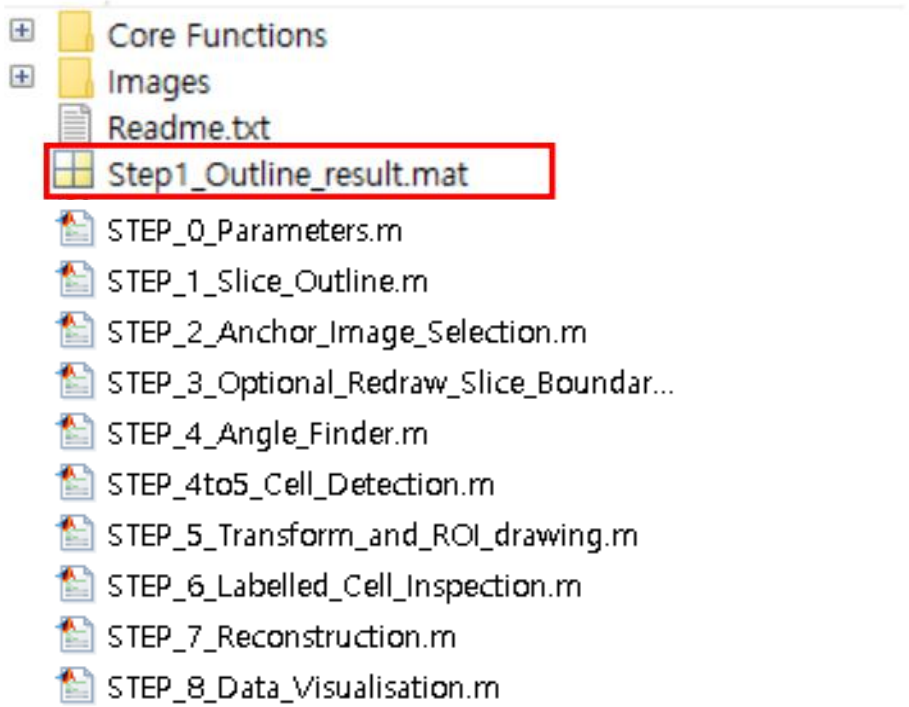


“Press the Space Bar and manually draw a rough boundary around slices”

Images appearing on these figure windows are edge-boosted

STEP_1_Slice_Outline

- Having finished running this step, a file—”Step1_Outline_result.mat”—containing slice boundary in formation of each brain slice will be saved. This file contains the slice boundary information of individual brain slice images

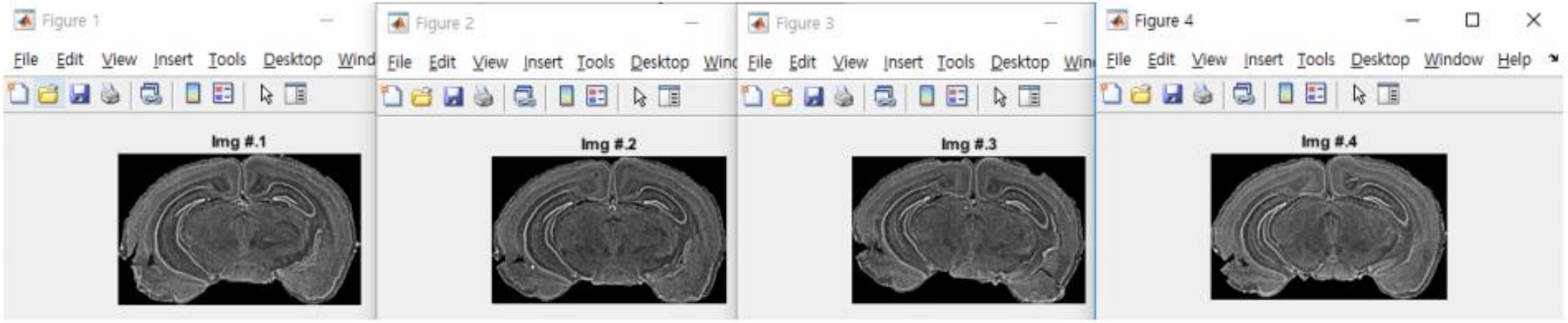


STEP_2_Anchor_Image_Selection

- In this step, you will
  1. **Sample "anchor images"** to find the slicing angles of the brain & AP positions of each slice
     - the AP positions of unsampled non-anchor images are interpolated between the anchors

  2. **Inspect the automatic slice boundary detection results** performed in Step_1 and choose images the boundaries of which must be redrawn

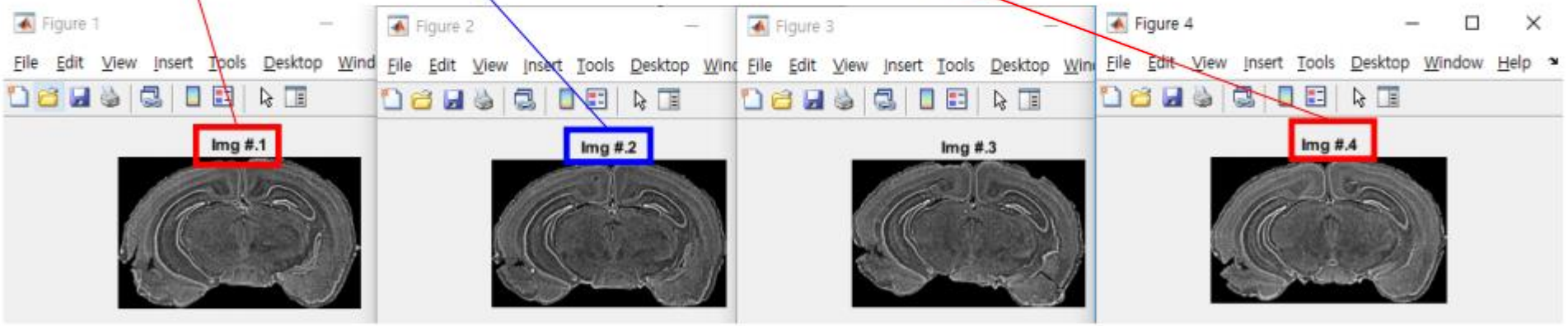
- In the command window, type "STEP_2_Anchor_Image_Selection" and press Enter; Image figure windows will pop up



STEP_2_Anchor_Image_Selection

- Once all image figures have appeared,
  double-click on the “STEP_0_Parameters.m” and open the script
- Type the indices of images sampled as “anchors”
- (Optional) If you find any images with weird boundaries drawn in Step_2,
  Put the indices of such images to re-draw their slice boundaries.
  If everything is fine, leave it empty (“img_IDs_reBoundary=[]”)

```
%% 3. Anchor Image IDs
anc_img_IDs= sort([1 4]);
img_IDs_reBoundary=[2];
threshold_scale = 1.5;
```



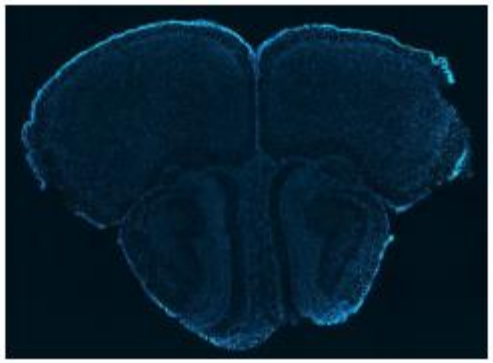
STEP_2_Anchor_Image_Selection

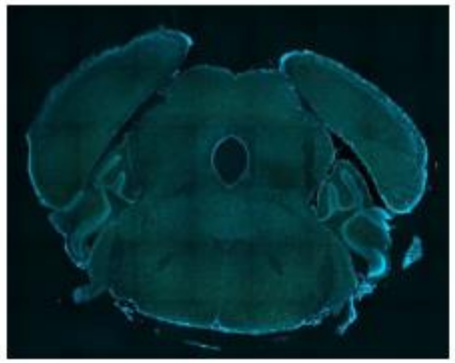
ImPoRtAnT WHEN CHOOSING ANCHOR IMAGES:

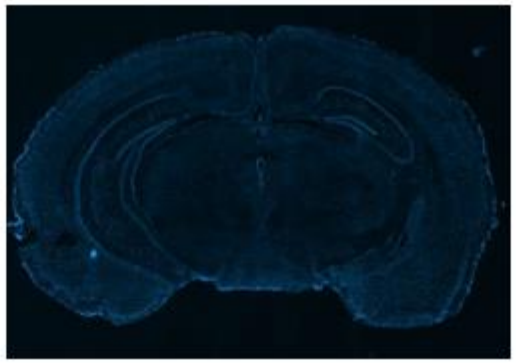
1. When sampling your anchor images, make sure that their qualities are good enough

➔ Not-so-great ones will hinder you from finding the correct slicing angles and make you sad

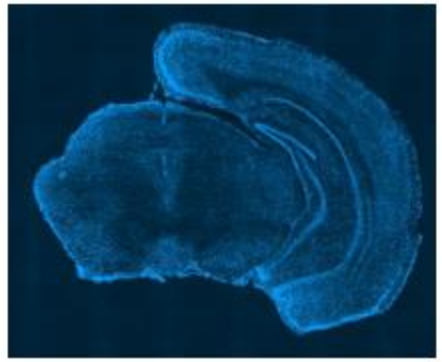
Good







Bad



STEP_2_Anchor_Image_Selection

ImPoRtAnT WHEN CHOOSING ANCHOR IMAGES:

2. The AP positions of **non-anchor images** will be **INTERpolated** between the anchor image positions

Say, if you have Images #1,#2,….#7, and set Images #2 & #6 as anchors,
Images #1 and #7 will not be further processed in the later stages,
because their AP positions can’t be interpolated

(Optional) STEP_3_Optional_Redraw_Slice_Boundary

- This is an OPTIONAL step to redraw the slice boundaries that you are unhappy with
  ➔ Change the “threshold_scale” value in the “Step_0_Parameters” script

```
%% 3. Anchor Image IDs
anc_img_IDs= sort([1 3]);
img_IDs_reBoundary=[2];
threshold_scale = 1.5;
```

- Run the code, and you will be asked to draw a rough slice boundary;

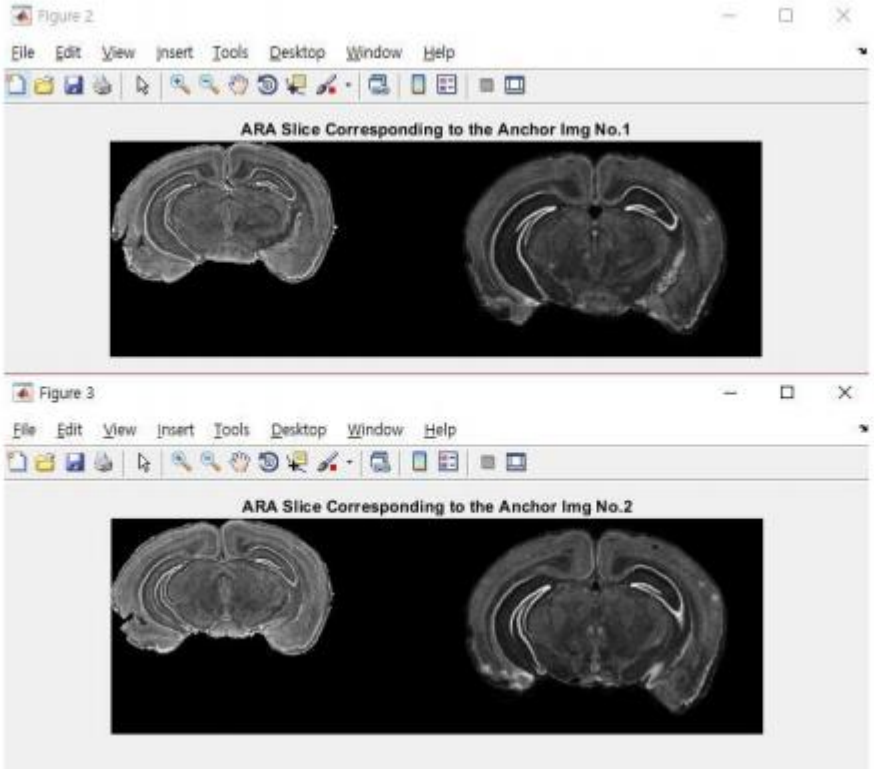
Example Image #2

Binary masks of an example image with varying threshold values

threshold_scale = **1.5** **0.1** **5**

STEP_4_Angle_Finder

- The AP positions of the anchor imgs & the slicing angles of the brain are found in this step
- In the command window, type “STEP_4_Angle_Finder”



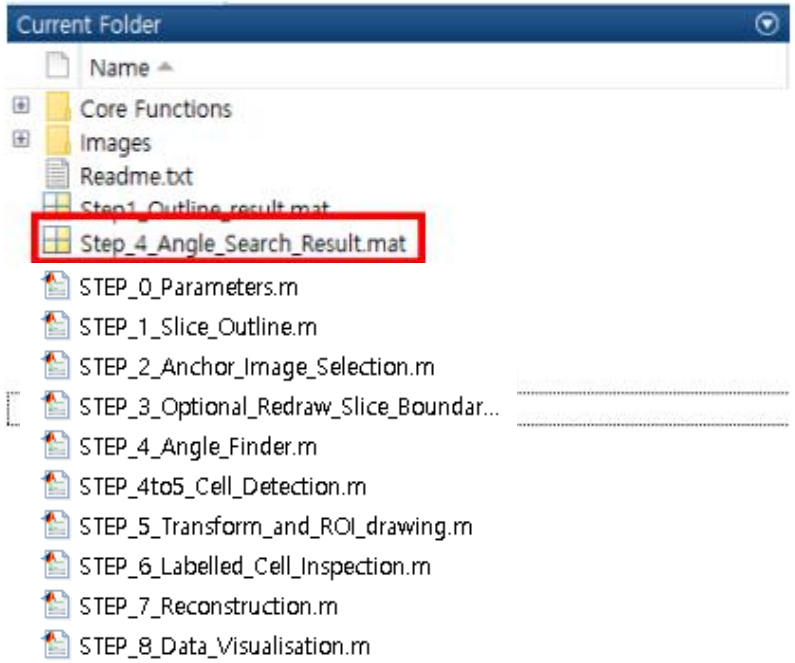
“Experimentally obtained imgs & corresponding Allen slice”

```
>> [yaw_stage5_max pitch_stage5_max]

ans =

     6     4
```

Output: The AP Positions of the anchor imgs & slicing angles (yaw & pitch) of the brain



STEP_4to5_Cell_Detection

- In the command window, type “STEP_4_Angle_Finder” (or run the script in the editor)

- In this step, labeled cells in each slice image are automatically detected

  - This step uses the script “**SomaDetection0827.m”** in the “Core Functions” folder for cell detection

  - **You are free to edit this code** for your own purposes:

    e.g. a CNN-based cell detection module

- If you are using the function that we included in the package,

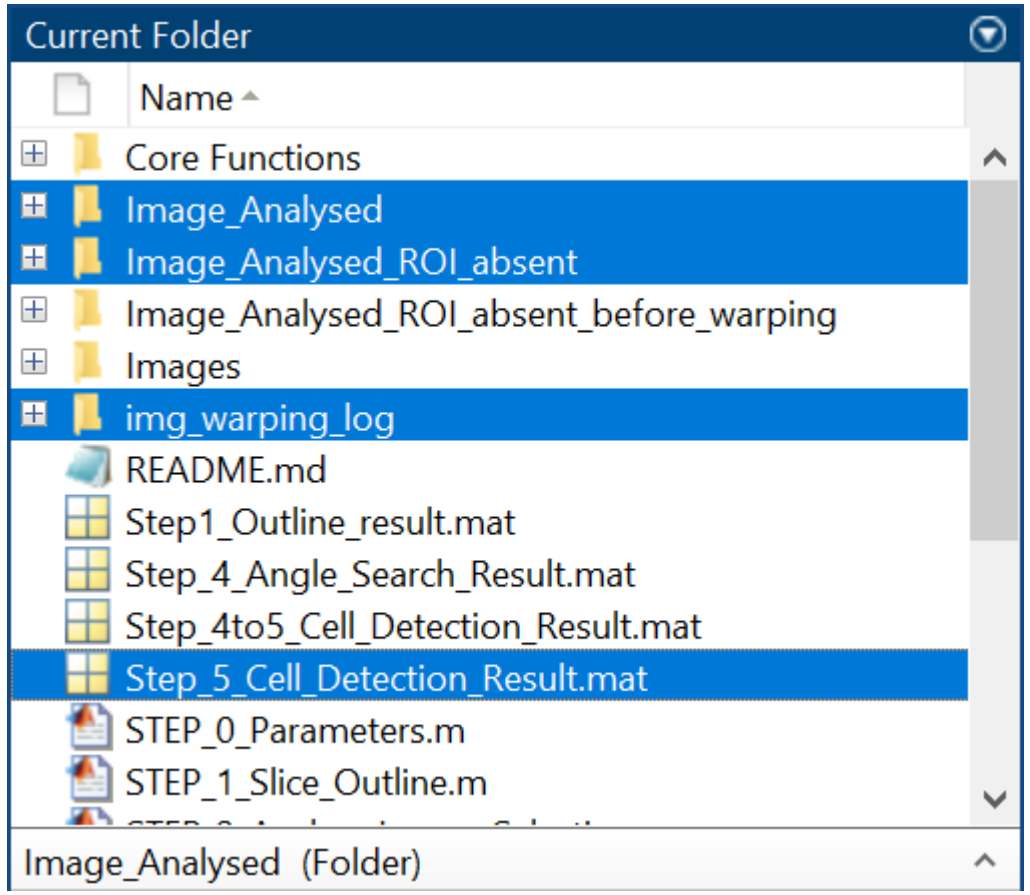
  you may need to adjust parameter ‘**cell_det_thresh**’ in “STEP_0_Parameters.m”

- When finished, “Step_4to5_Cell_Detection_Result.mat” that has the coordinates of the detected neurons in each slice will be saved.

  - Note that these coordinates are NOT the Allen-registered coordinates… not yet….

STEP_5_Transform_and_ROI_drawing

- This step registers experimentally obtained images to their corresponding Allen cross-sectional images & warps the cell coordinates accordingly.



→ Warped Imgs with ROI boundaries+Cells detected

→ Warped Imgs without ROI boundaries+Cells detected

→ Elastix logs for image registration

→ A struct containing **warped cell coordinates,** img AP positions, and img transformation parameters

STEP_5_Transform_and_ROI_drawing

- If you wish to adjust the Elastix settings, refer to

  Core Functions → elastix_functions → warping_parameters_Affine.txt

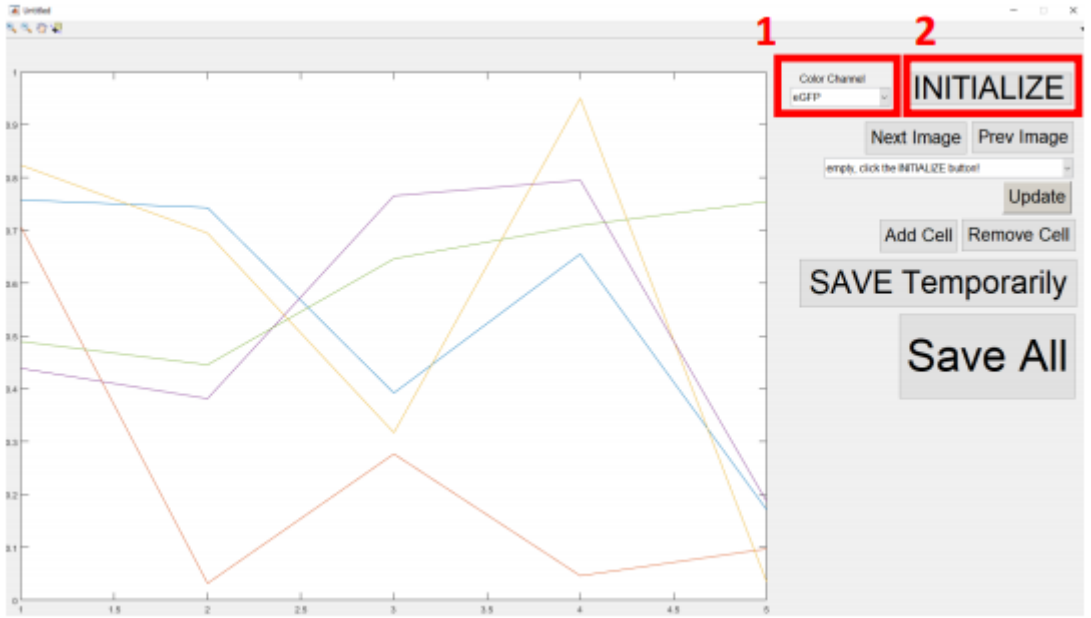
  warping_parameters_BSpline.txt

(OPTIONAL) STEP_6_Labelled_Cell_Inspection

- You may wish to
  1. remove false-positive neurons detected
  2. manually mark the neurons that are not automatically detected

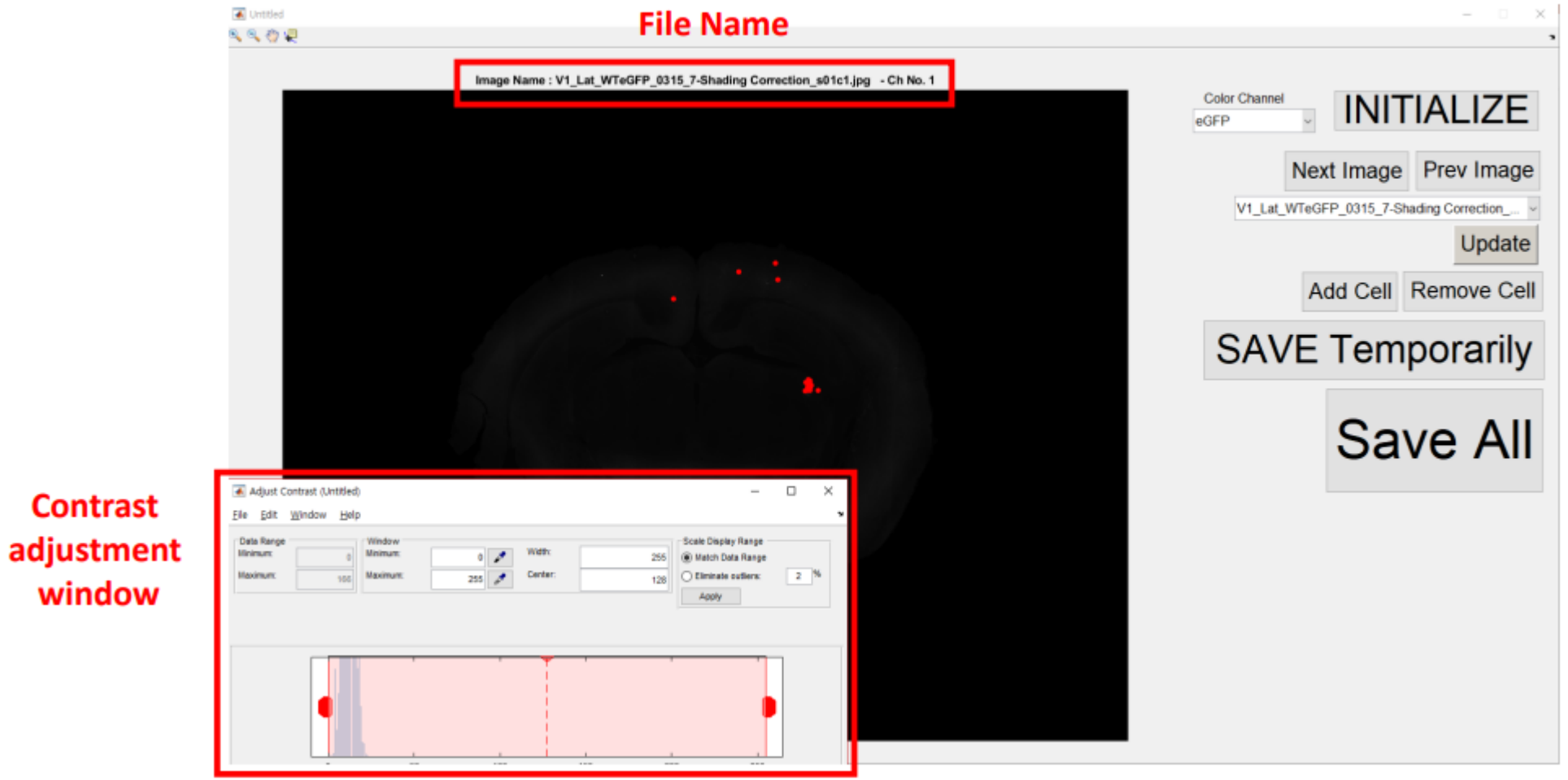
- In the command window, type “STEP_6_Labelled_Cell_Inspection”

➔ A GUI window will appear
➔ Set the color channel that you’d like to inspect (if multi-color) and Click on the “INITIALIZE” button



(OPTIONAL) STEP_6_Labelled_Cell_Inspection

- The first image of the ARA-matched image set will appear

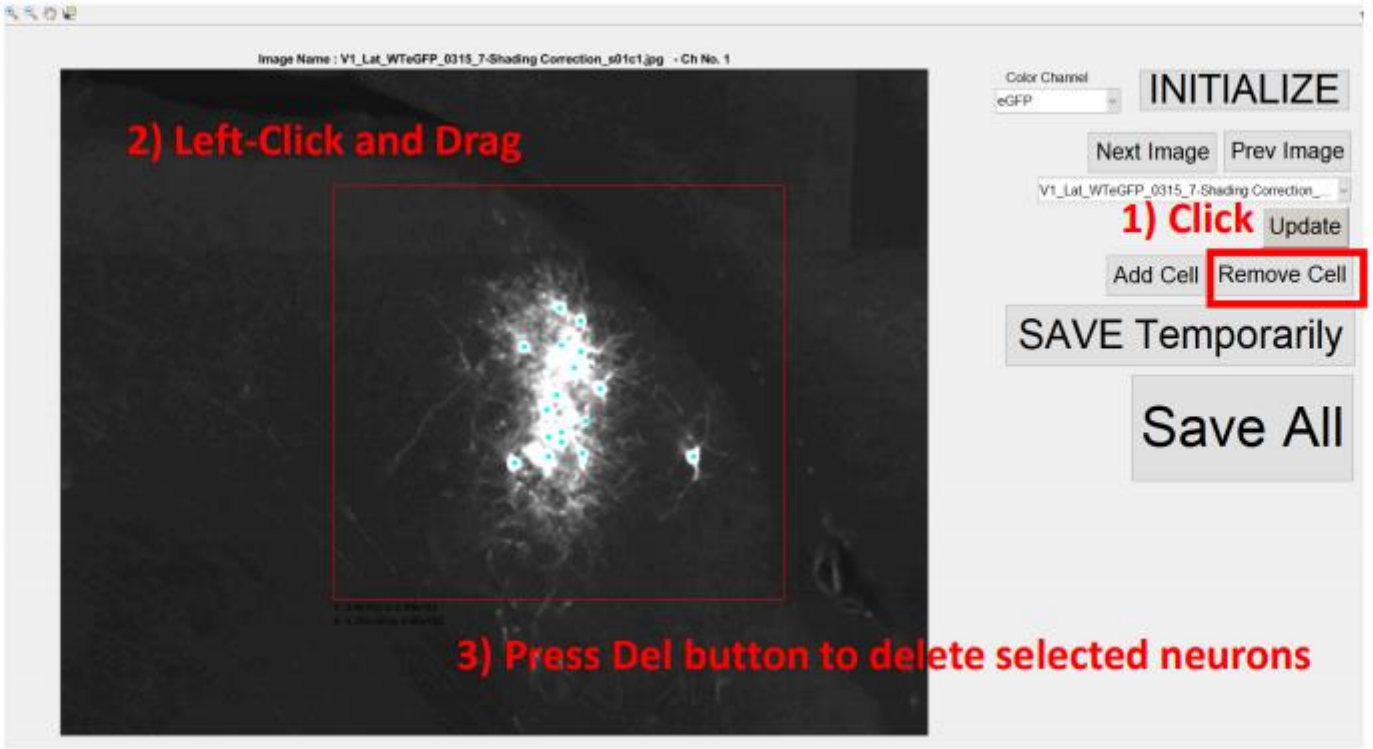


(OPTIONAL) STEP_6_Labelled_Cell_Inspection

- To add neurons,
    1) Click on the “Add Cell” button in the GUI panel
    2) Left-click on the neurons that you want to mark
    3) Right-click if you are happy with adding new (false-negative) labeled neurons

(OPTIONAL) STEP_6_Labelled_Cell_Inspection

- To delete neurons,
    1) Click on the “Remove Cell” button in the GUI panel
    2) Left-click & drag window the cells you want to “undetect”
    3) Press “Del” on the keyboard to delete



(OPTIONAL) STEP_6_Labelled_Cell_Inspection

- After inspection,

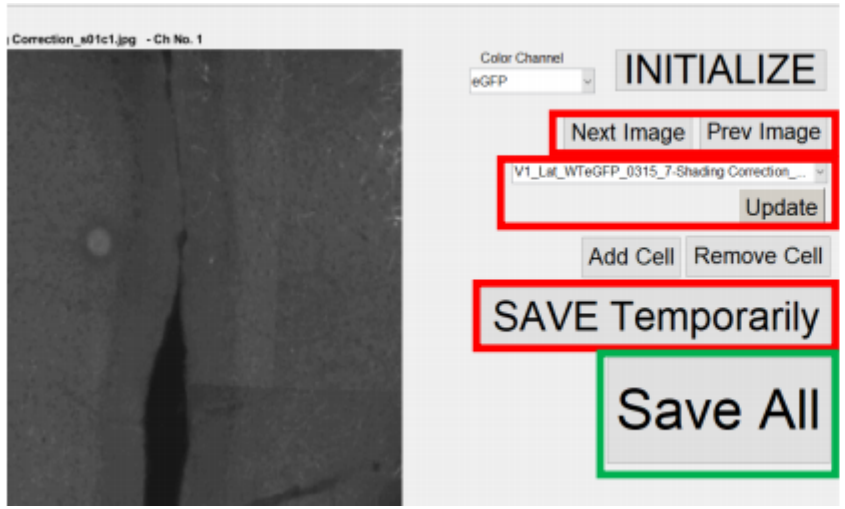
  1) Click on “Save Temporarily”; otherwise the edited results will not be updated

  2) Click on “Next Image” or “Prev Image” to move on

  - You can also select your image from the dropdown menu
    and click on the “Update” button

  3) When you are done with all the images, click the “**Save All**” button.

  - Otherwise, your effort will not be saved, and you will be sad.
    I have been sad quite a few times



2) Click Next Image / Prev Image to move on

2b) Choose the image in dropdown menu & click Update button

1) Click – otherwise the edited result will not be updated

3) Press “Save All” to update and save all neuron data

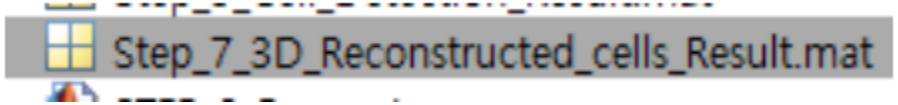
STEP_7_Reconstruction

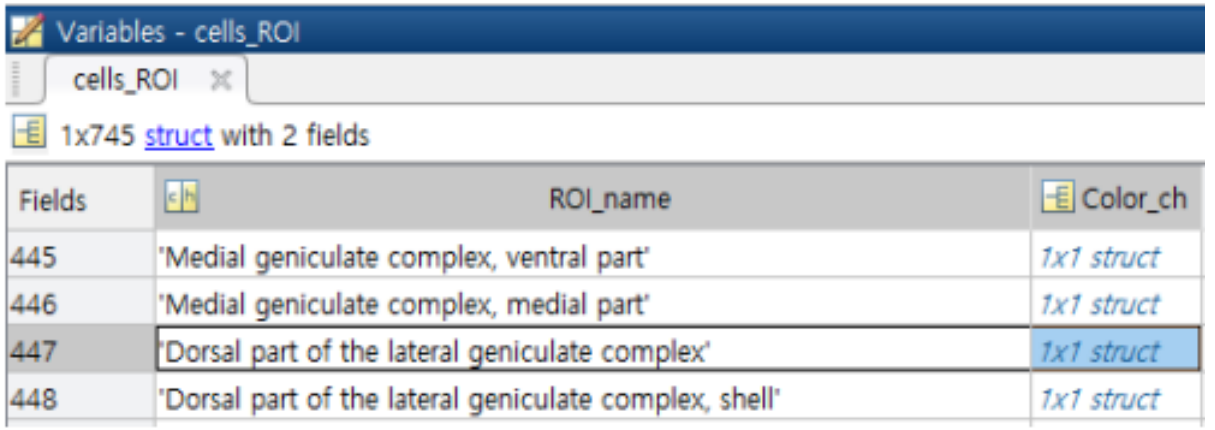
- Run the “STEP_7_Reconstruction” script.
  - This script annotates detected neurons to their ROIs

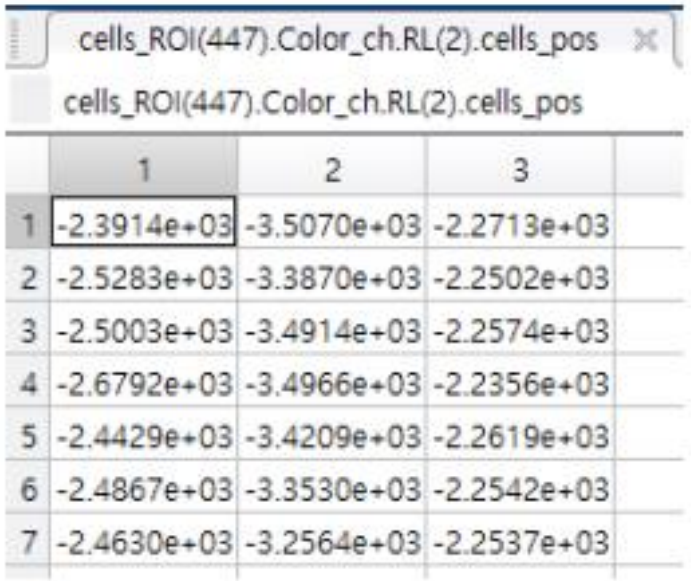
STEP_7_Reconstruction

- (Optional) If you need to access the position coordinates of detected neurons in ARA,

1. Load the saved .mat file
2. Navigate to the ROI that you are interested in
3. You'll see the 3-d coordinates of your detected cells
   1st, 2nd, and 3rd Columns = ML, DV, AP coords (um) , resp.

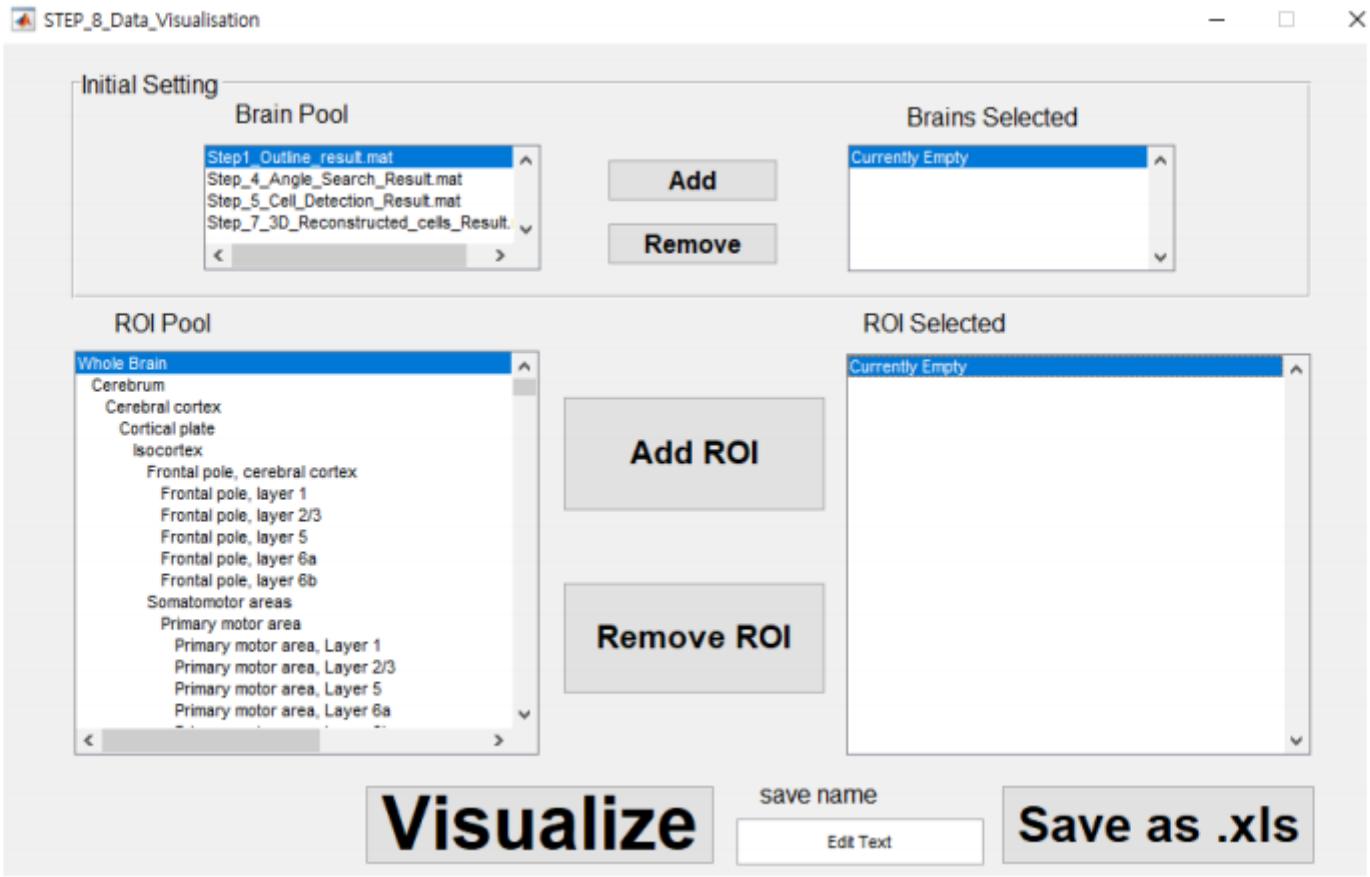






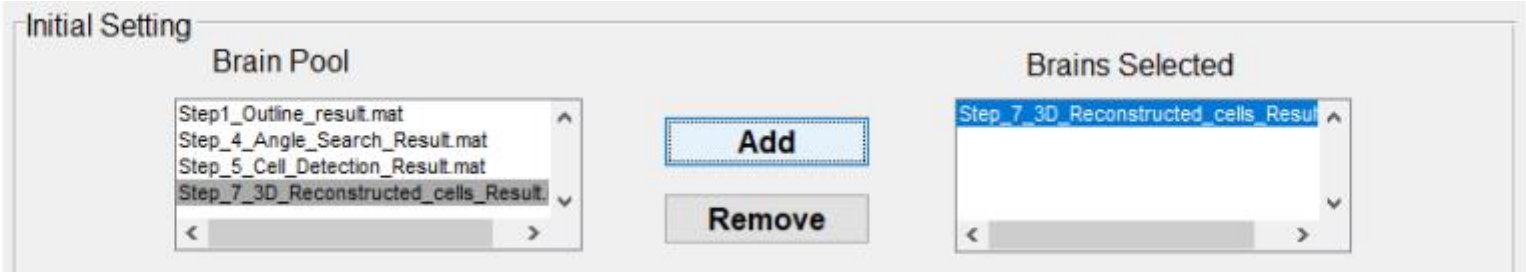
STEP_8_Data_Visualization

- Run the “STEP_8_Data_Visualisation” script.
  → A GUI window will appear



STEP_8_Data_Visualization

- Add the result from Step7 that you wish to have a look in 3d.



*** If you'd like to **compare the results from multiple mice**,

1. Change the result file names.

e.g. Mouse # 1's Step_7_3D_Reconstructed_cells_Result.mat ➔ Mouse_1_VISp.mat

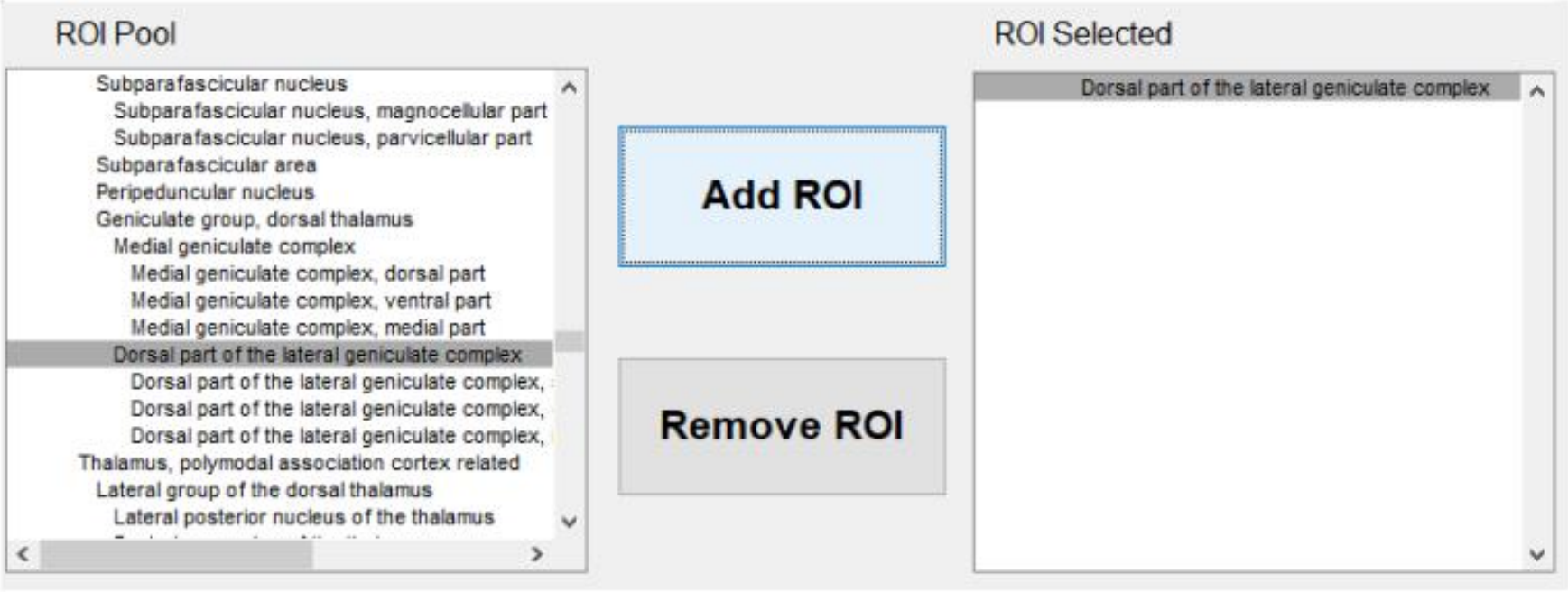
Mouse # 2's Step_7_3D_Reconstructed_cells_Result.mat ➔ Mouse_2_VISp.mat

2. Locate the name-changed files in a same folder

3. Run Step 8

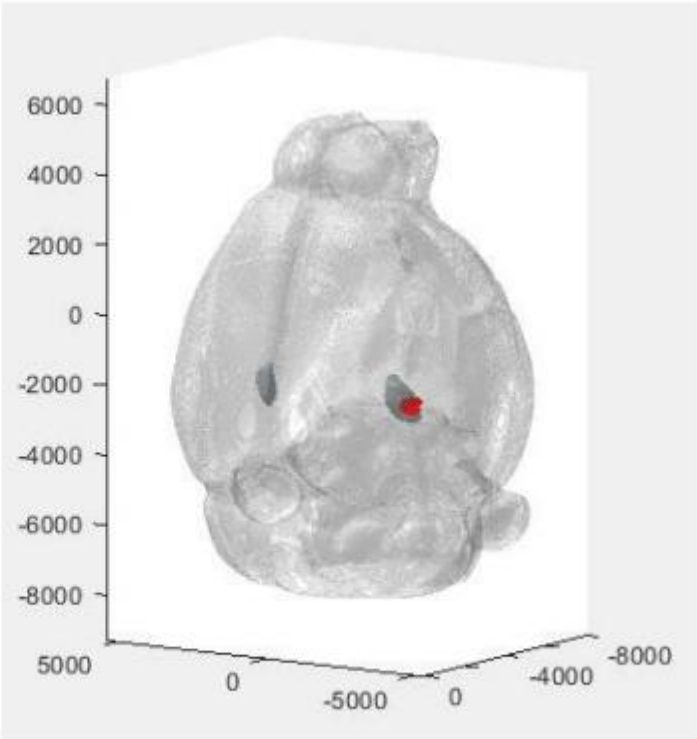
STEP_8_Data_Visualization

- Add the ROIs that you want to visualize



STEP_8_Data_Visualization

- Click on VISUALIZE button

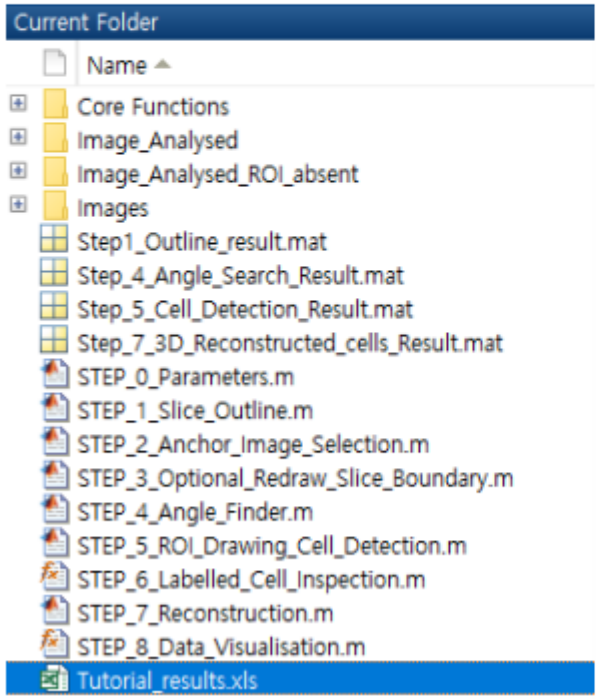


*** If you wish to modify the visualization settings,

Change the plotting options in Lines 284-372 in “STEP_8_Data_Visualization”

STEP_8_Data_Visualization

- If you wish to export your cell-counting results, type in the save name and click the "**Save as .xls**"



	A	B	C	D
1	ROI name		Step_7_3D_Reconstructed_cells_Result.mat	
2			Left hemi (cells)	Right hemi (cells)
3		Dorsal part of the lateral geniculate complex	0	55
4				
5				
6				
7				
8				

Troubleshooting

- Most of the errors were caused by uses' putting incorrect parameters settings

  Please re-check the parameters you have set in the "STEP_0_Parameters"

- If you are coming back after working on other stuff with MATLAB, (e.g. you finished working through Step 0 to 6; you now want to work on Step 7) and MATLAB gives you an error mentioning something like "directory, no such file found"

  1) Go to the "mother" directory that you have made in Slide # 5

  2) Run STEP_0_Parameters

  3) Re-run your script (e.g. STEP_7)

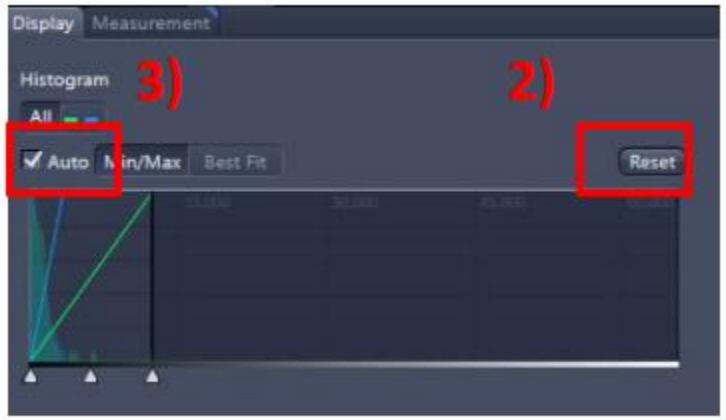
- If you still get errors after trying the aforementioned measures or have any other queries,

  contact the corresponding author, Prof. Se-Bum Paik (sbpaik@kaist.ac.kr)

Tips if you use AxioScanner (Zeiss)

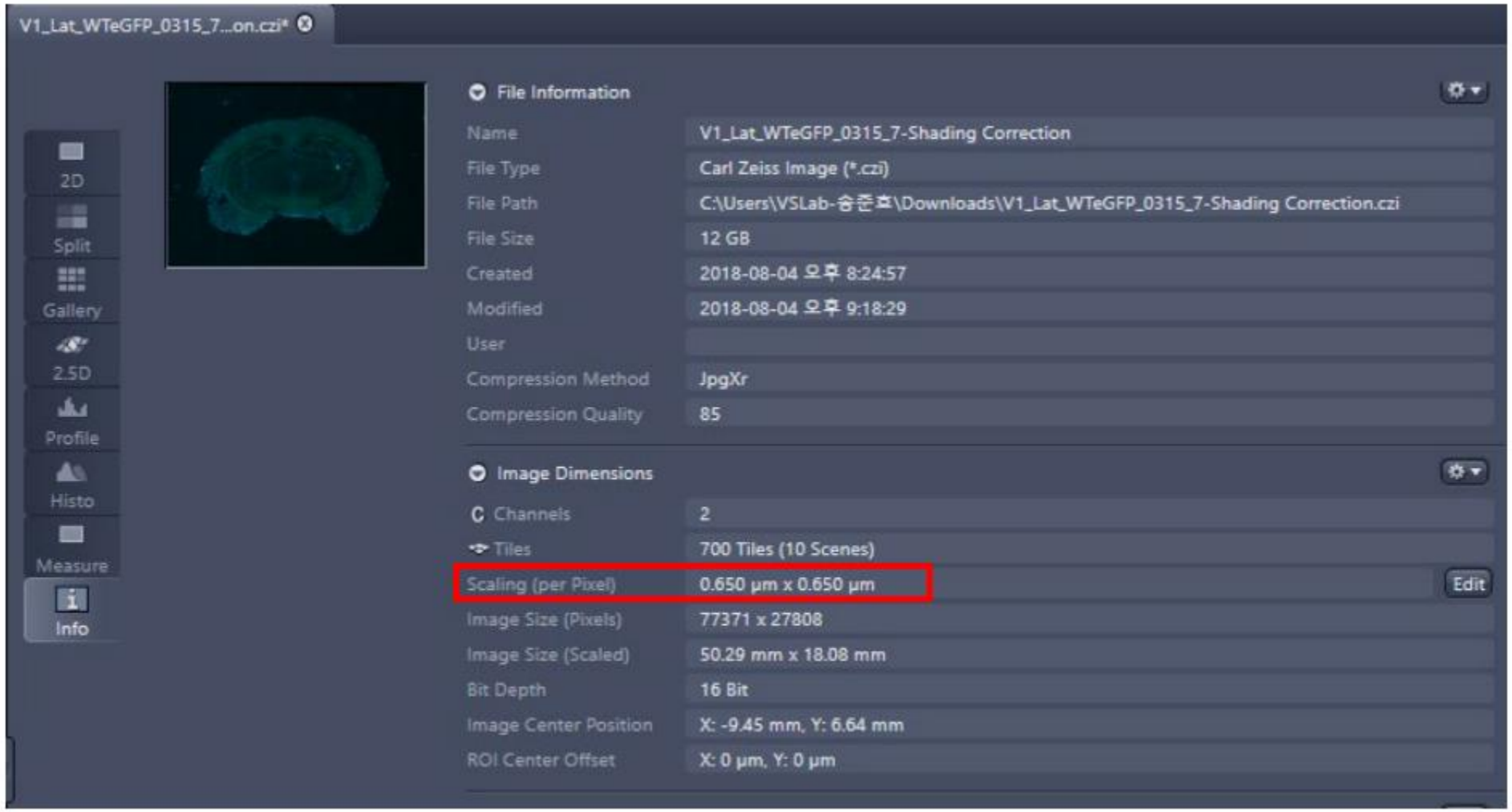
- We have done the following steps when exporting slice images using ZEN

1)



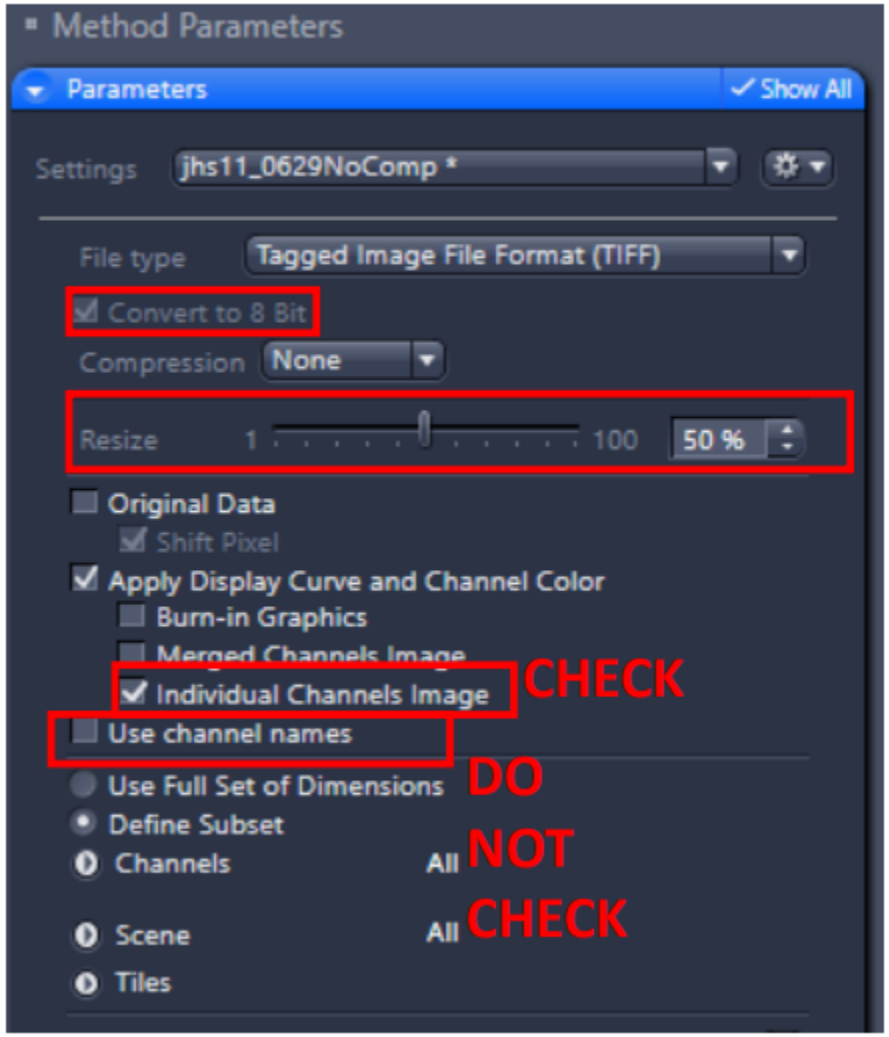
1. Intensity Normalization
   - Check the "Scene" box
   - Click on the Reset button
   - then check the "Auto" box
   - Save the . czi file (ctrl+s for Windows)
     → Intensity normalized for each slices

Tips if you use AxioScanner (Zeiss)



2. Check your image metadata → **Check parameters in STEP_0_Parameters**

Tips if you use AxioScanner (Zeiss)



3. Export Settings
   - Convert to 8-Bit
   - Optional (Resize factor = 50%)
     → This will downsample images by 50%
     → In this case, the xy_pix should be doubled
     i.e. xy_pix = 0.650 *2 in STEP_0
     → This will speed up the computation

   - Please check “Individual channel images “
   - Please DO NOT check “Use channel names”