

Brain Stitcher

Northeastern University | Boston, MA, USA

User Manual Version 0.1

Seyed Mostafa Mousavi Kahaki & Armen Stepanyants

2017

Contents

[Introduction 2](#_Toc502137348)

[System requirement and installation: 2](#_Toc502137349)

[BrainStitcher workflow 3](#_Toc502137350)

[Feature extraction 3](#_Toc502137351)

[Feature matching 3](#_Toc502137352)

[Global optimization 3](#_Toc502137353)

[Resampling 3](#_Toc502137354)

[Code structure 4](#_Toc502137355)

[Input file 5](#_Toc502137356)

[Output files 6](#_Toc502137357)

[GUI sections 7](#_Toc502137358)

[Section 1. Menu and toolboxes 7](#_Toc502137359)

[Section 2. Input file 7](#_Toc502137360)

[Section 3. Parameters 8](#_Toc502137361)

[Section 4. Running controller 9](#_Toc502137362)

[a. Process controller: 9](#_Toc502137363)

[b. Parallel processing: 9](#_Toc502137364)

[Section 5. Control buttons 10](#_Toc502137365)

[Section 6. Visualization controller 10](#_Toc502137366)

[Section 7. Dataset map 10](#_Toc502137367)

[Section 8. Log view 10](#_Toc502137368)

[Section 9. Visualization section 11](#_Toc502137369)

[Visualization 12](#_Toc502137370)

[Parameters description 12](#_Toc502137371)

[Steps to run the registration 13](#_Toc502137372)

[About 14](#_Toc502137373)

[Contact information 14](#_Toc502137374)

[Support 14](#_Toc502137375)

[Related publications and posters 14](#_Toc502137376)

[Contributions 14](#_Toc502137377)

# Introduction

The ability to map neural circuits on the scale of an entire brain is critical for advancing our understanding of brain functions. Circuit mapping can be based on whole-brain imaging of sparsely labeled populations of neurons with 3D confocal or two-photon microscopy. *BrainStitcher* is a software for stitching and registration designed for different proposes including 1. Stitching of 3D image stacks of entire brain, 2. registration of time-lapse images, and 3. registration of images within stacks.

1. **Stitching of 3D image stacks of entire brain:** whole brain imaging experiment, if applied to the mouse brain, would result in tens of thousands of image stacks, totaling several terabytes of data. Because, imaging is generally done with small overlaps between neighboring stacks, the information contained in the stack overlap regions can be used for stitching. The *BrainStitcher* software can provide the ability of stitching 3D stacks in both sequential and parallel and output the registered stack positions along with the transformation for each stack for further analysis.
2. **Registration of time-lapse images:** the second goal of this application is to register 3D stack images captured in different times. These images are usually having high overlap because they are taken from a same part of the brain in different time. The current *BrainStitcher* software can also register this type of images with a high accuracy.
3. **Registration of images within stacks:** the *BrainStitcher* software can register stack slices within a single stack which is an important task in Neuroscience and many other fields. Large scale image stacks captured in live animal can be transformed within the slices of the stacks. This is a common issue specially in EM data. Using this ability of the software, you can register the slices based on different transformation provided in the GUI.

# System requirement and installation:

*BrainStitcher* is designed to run on both Windows and Unix base systems with Matlab installed package (preferable with Matlab 2017b). To install the *BrainStitcher*, download the BrainStitcher.zip file from <http://www.northeastern.edu/neurogeometry/resources/BrainStitcher/> and extract its content. Install the software using the installation file and open on your system.

# BrainStitcher workflow

The *BrainStitcher* software contains four different steps including feature extraction, feature matching, global optimization, and resampling. We described each step and their important parameters here.

## Feature extraction

Features are small volumes centered around local maxima in an image stack. To detect feature centers, we first apply the Laplacian of Gaussian filter (LoG) to smooth out image intensity. You may choose other filters from the GUI section 3 (please refer to the GUI sections of this manual). Next, we identify the center of the first feature with the maximum intensity in the filtered image. Subsequent feature centers are associated with the intensity maxima after the exclusion of all previously found features. This process can be done in parallel if your machine is supporting that. You may choose the parallel execution in the section 4 of the GUI (please refer to the GUI sections of this manual).

## Feature matching

In the second step, we find the matched feature points in neighbor stacks. for this goal, the feature matching process performed in two steps. In the first process, we do coarse feature matching. For this goal, for all overlapping stack pairs we compute a matrix of feature similarities. This matrix is used as an input to the Hungarian algorithm [1] to establish initial correspondences between features. In the next process, the final matched points detected using a step called refinement of initial matches. Initial matches of features may contain outliers. These outliers are eliminated by using RANSAC algorithm [2] coupled with an optimal 3D transformation of source feature positions and comparison of the result with the positions of target features. We have implemented and tested the following transformations: translation, rigid, affine, and custom non-rigid. You may choose the desired transformation in the section 3 of the GUI (please refer to the GUI sections of this manual). This process can be done in parallel if your machine is supporting that. You may choose the parallel execution in the section 4 of the GUI (please refer to the GUI sections of this manual).

## Global optimization

After we find the matched feature points between each stack pars, we need a global optimization process to find the optimum global position and transformation for each individual stack in the space. This process is done based on optimum global translation, optimum global rigid, optimum global affine, and optimum global non-rigid functions. You may change the parameters related to this section in the parameters.m file. For more information regarding the parameters, please check the *Parameters description* section of this manual.

## Resampling

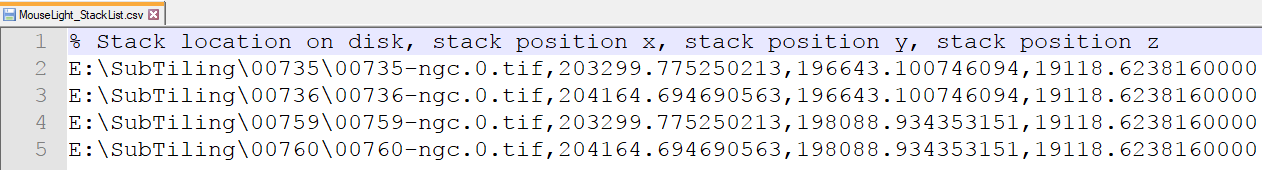
We almost done. Global optimization process provides the final positions of each stack in the 3D space and a transformation matrix for each stack. In the resampling step, we create non-overlap tiles which the global positions and transformation are applied to them. These non-overlapped tiles are registered and ready to use for further analysis. You may choose the preview and the stack you want to view from section 4 of the GUI (please refer to the GUI sections of this manual). You may change the parameters such as tiles size in the parameters.m file. For more information regarding the parameters, please check the *Parameters description* section of this manual.

# Code structure

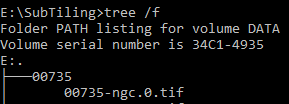
*BrainStitcher* requires stack tiff files (3D tiles) and a csv file indicating the location of the files and the stack positions (outputted from the microscope) as initial stack’s locations.

# Input file

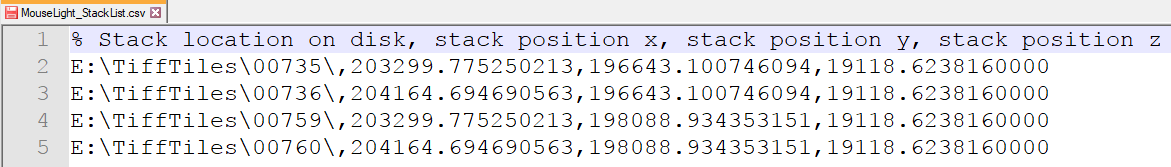
The main input of the software is a “.csv” file including the stack locations on disk and stack positions (x,y,z locations captured from microscope). In the input csv file, each line(row) indicates one stack or tile. The first line of the csv as presented in Fig. 1(a) can be a comment line. Each stack files (for example 00735 with size 1536×1024×251) can be a single tiff file or can be 251 files of size 1536×1024. If all layers of a stack are stored as a single tiff file, you need to specify the location and the name of the file of the stack as can be seen in the following figure.

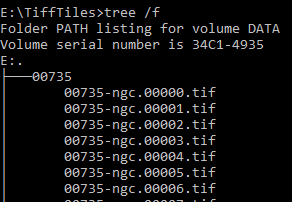


In this example, which presented in the right image, the stack *00735* (1536×1024×251) is stored as a single file named 00735-ngc.0.tif. The structure of the folders is not important here while we include the full path of the file in the input “.csv” file.



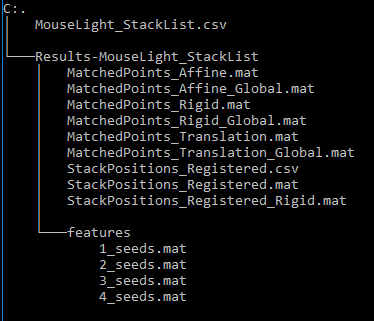
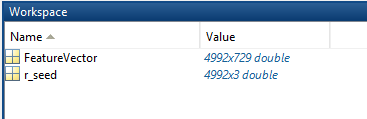
You also may have separated files for each tile. In this case each tile (for example 00735 with size 1536×1024×251) is stored as 251 tiff files with size 1536×1024 in a folder named *00735.* In this case, you only need to include the folder address of each tile in the rows of the .csv file as presenter in the Figure below.

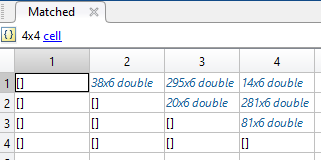




In this example, which presented in the right image, the stack *00735* (1536×1024×251) is stored as 251 tiles with size 1536×1024 with sequential names in a folder named 00735. The structure of folders is important here while each folder should only contain files from one stack. The sequence of the file names also should be based on the numbering of the slices.

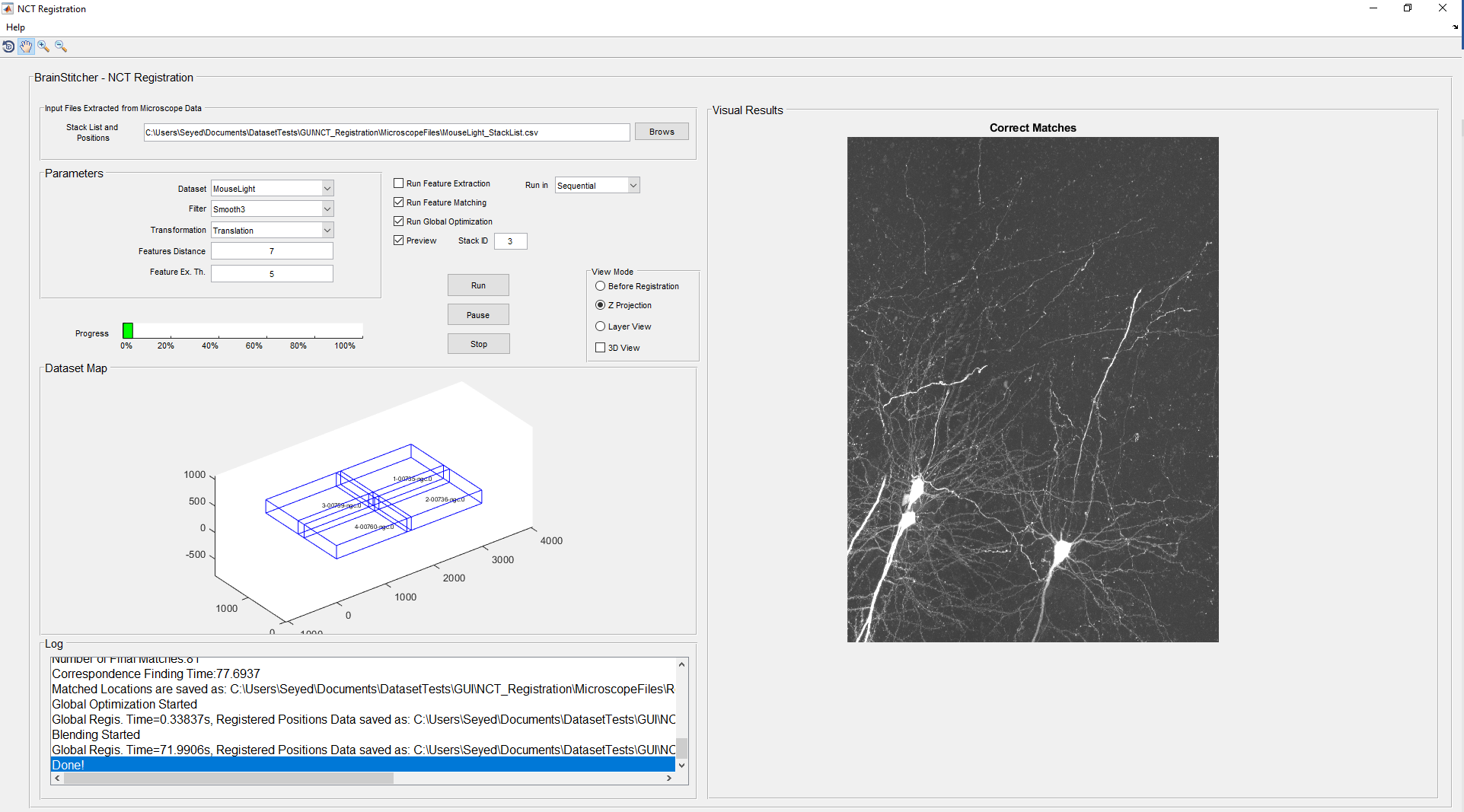
# Output files

In this section, we describe the output file and folders structure along with the content of the output files. After you select the input *csv* file (in example MouseLight\_StackList.csv) and click on the *run* button, the registration process including feature extraction, feature matching, and global registration will start based on the parameters you have chosen. Based on the csv file name, a result folder will be created as shown in the image (in example Result\_MouseLight\_StackList). The feature files will be stored in a folder called *features*, and there be some result files in the main folder. As shown in the right image, here we have four feature files in the features folder indicating we have four stacks in the list. The name of each feature file is take from the stack name if the stack is a single file, or stack id if the stack is saved as several images. Each features file includes two variables named FeatureVector and r\_seed. The r\_seed store the feature positions in x,y,z within the stack and each row of the FeatureVector is the feature vector extracted from the feature neighbors intensities. In example, here we have 4992 features and 729 columns corresponds to 9×9×9 neighbor intensities of each feature.

The next category of result files is related to the feature matching process. In this process, for each pairs of stacks we need to have a set of corresponding points as matched features. The matching result using translation process are saved as MatchedPoints\_Translation.mat, and the result of matching using affine transform are saved as MatchedPoints\_Affine.mat, and so on. Each matched pint file, contains an upper triangular cell array were nonempty cells stored the matched points for each stack pairs. For example, in this example, the first row and second column includes a matrix of 38×6 which are the corresponding feature positions between stack with ID 1 (the firsts stack in csv file) and stack with ID 2 (the second stack in csv file). Therefore, we have 38 corresponding point between stack 1 and 2. The second dimension is 6 which indicate three coordinates of the features from stack 1 and three coordinates of features from stack 2.

# GUI sections

After install and run the application, an GUI (as shown below) will be open which can control the process of registration. In this section, we describe different sections of the GUI. For this goal, we created 9 different section as presented in the following image.



**1**

**8**

**7**

**6**

**5**

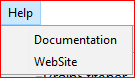
**4**

**3**

**9**

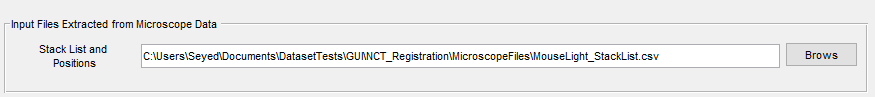
**2**

## Section 1. Menu and toolboxes

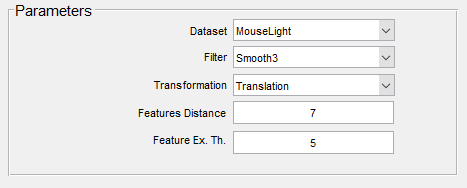
  Menu and toolboxes are the first section of the GUI we are going to describe here. In the help menu, you can access to the Documentation and Website links. In the toolbox section, four different tools are available including rotation, hand, zoom in and zoom out tools. In order to be able to use them, you need to click on the result image in the visualization section (section 9) and start to use these tools.

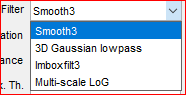
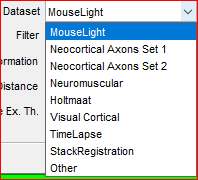
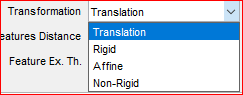
## Section 2. Input file

In this section, you can see the default input file and shown below. You may click on the *Brows* button to select your input file. For more information regarding the input file structure, please check the *Input File* section.



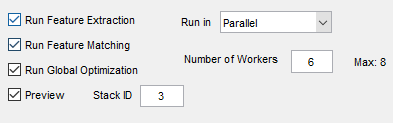
## Section 3. Parameters

You can change some important parameters from the GIU, while others are defined in the parameters.m file as described in the *Parameters* section along with their default values and potential affects. The parameters which are available in the GUI are: Dataset, Filter, Transformation, Feature distance and Feature Extraction Threshold. In the Dataset dropdown menu, there are some available dataset which you may choose from the list. For other datasets, you may choose *other* and brows your own dataset input file.

## Section 4. Running controller

### a. Process controller:

The registration process has four main steps including feature extraction, feature matching, global registration and resampling (preview). You can choose which process you want to run from the running controller section. Please note that, ruing preview required information from Global registration process, registration process requires feature matching files, and feature matching requires available features. This section helps you to repeat a specific process if its required information is available. In example, if you already run the feature extraction and the features are good enough, you can unselect the *Run Feature Extraction* section to save the time and runs only other sections.

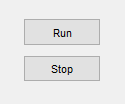
**b**

**a**

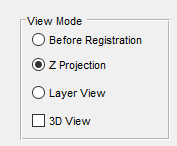
### b. Parallel processing:

The software is designed to run in parallel. In this section, you may choose to run the process in sequential or parallel. You can choose the number of workers based on your system cores. The *Max* is the maximum number of available cores provided by your system, which you may not exceed that number when you choose the number of workers. Please note that, running in parallel will affect the progress bar and visualization section which means the progress showing by progress bar is not accurate. Moreover, during the run in parallel, the feature extraction and matching results are not available and you only can see the result after the process is completed. Running I parallel is only recommended for high number of input stacks.

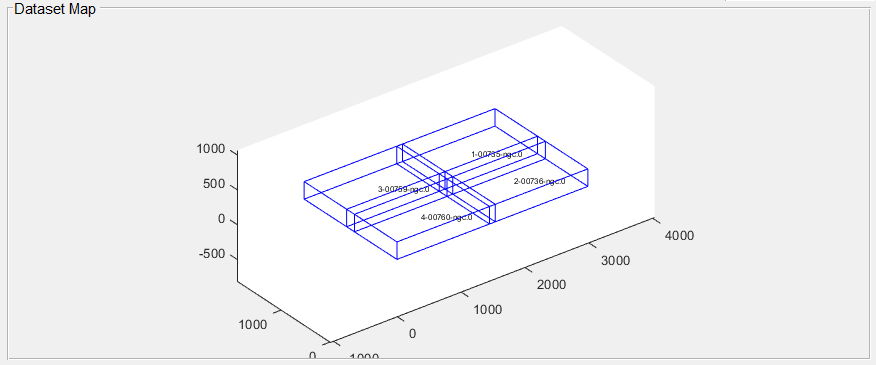
## Section 5. Control buttons

Simply you can run the process by clicking the *Run* button in this section when you set the input and the parameters (in GUI and parametes.m file). If you willing to stop the process, you may click the *Stop* button to stop the process. Stop process my not affect instantly because the process need to be completed at some procedures.

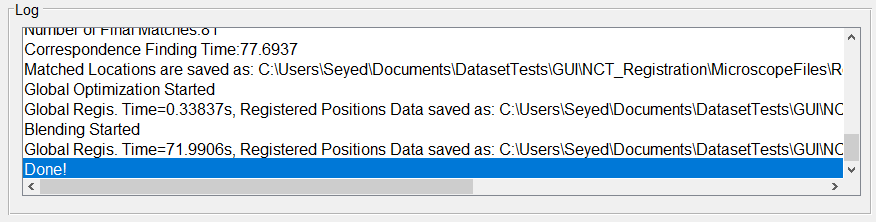
## Section 6. Visualization controller

This section controls the registration visual results. When the registration process completed, you will be able to see the stacks (based on the stack you have choose in section 3(a) and its neighbor stacks). You may choose *Before Registration* which will show the overlapped stacks before registration using microscope files provided in the input *csv* file. For visualizing the stacks after registration, you may choose *Z Projection* or *Layer view*. You are also able to see the 3D view along with other options. For more information regarding visualization, please refer to the *Visualization* section of this user manual.

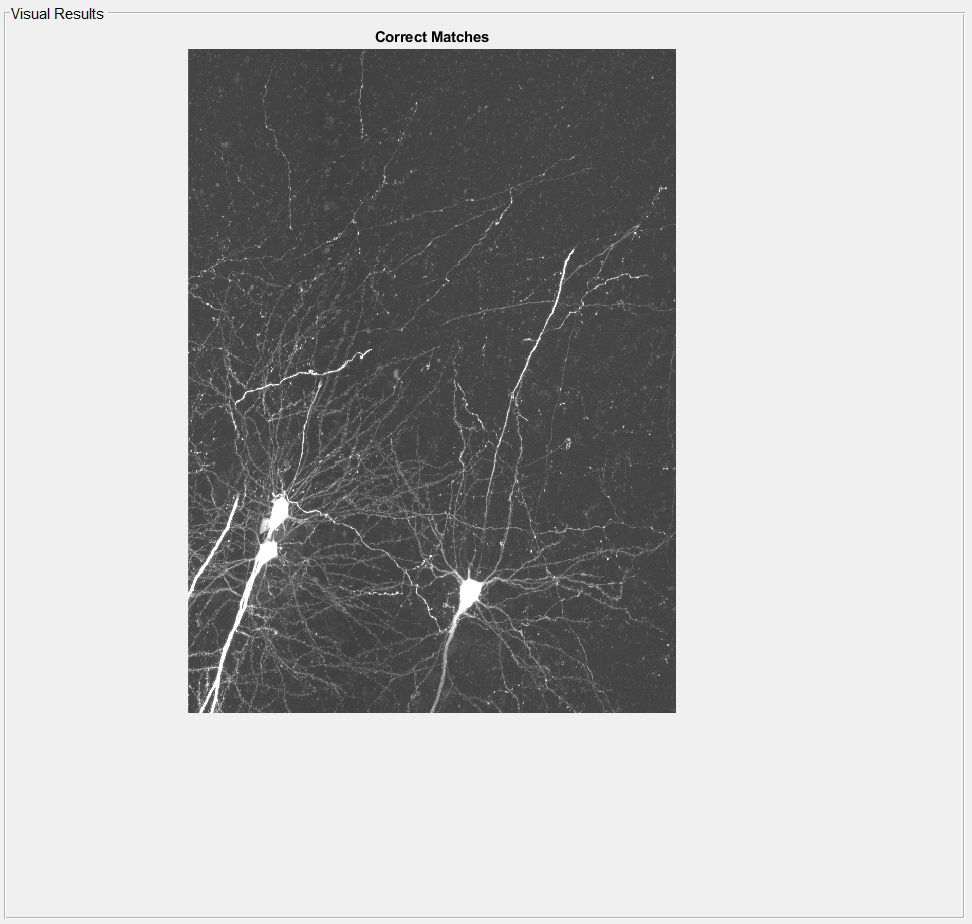
## Section 7. Dataset map

In the Dataset Map section, you can visually see the dataset 3D view based on the stack sizes and location provided from the microscope along with each stack ID and filename. To rotate, zoom in and zoom out, you can click on the 3D view and use the toolboxes of the GUI. The text inside each stack is the stack ID (the first number) and the name of the stack separated by a dash (-).

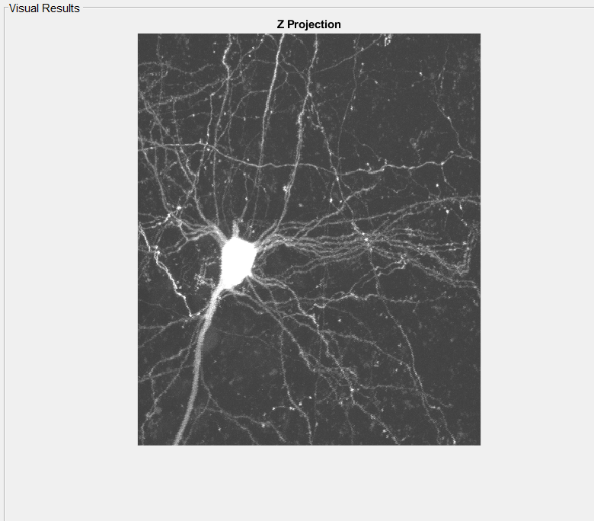
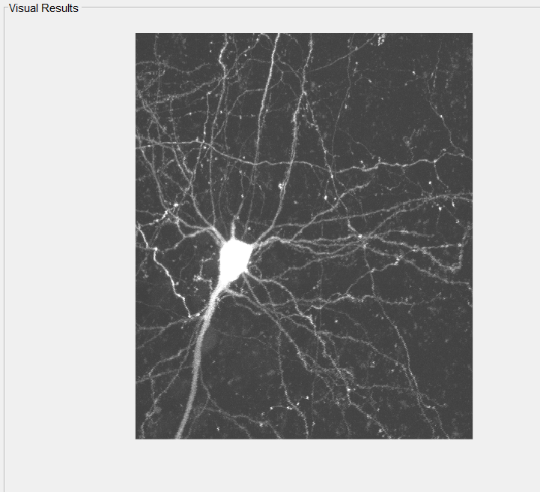
## Section 8. Log view

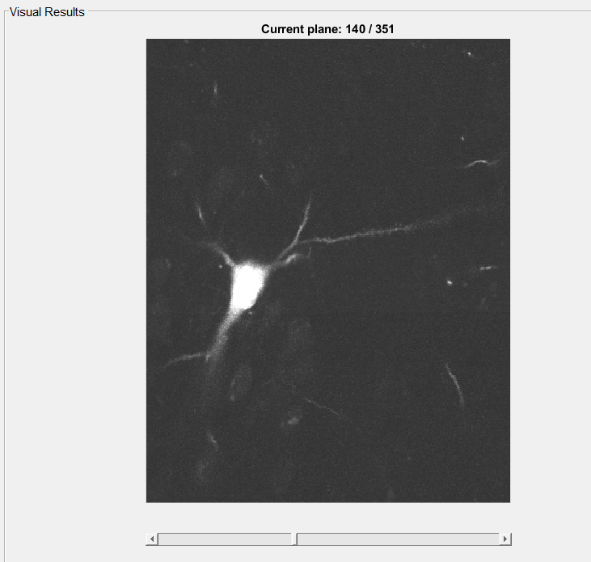
During the registration process, you can see the progress by following the text in the Log section of the GUI. This section also shows the number of extracted features, number of detected correspondence features with timing of each process. The location of the result files is also listed in this section after each process completed.

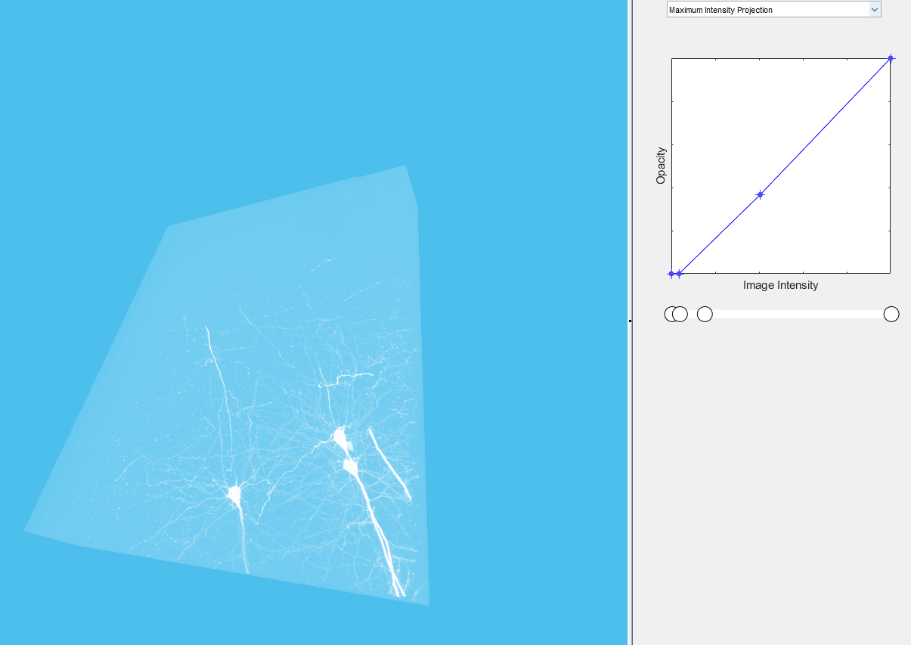
## Section 9. Visualization section

The result of feature extraction, feature matching and resampling will present in this section after completion of each step. If you are running the process in parallel, this section will only show the final resampling results. If you choose *Layer View* option in section 6 of the GUI, you will be able to see a scroll bar to scroll through z planes of the result. To activate the scrolling or toolboxes, please make sure that you click on the image. To save the result of registration, please set the *params.BP.remove\_pad* to one so the software will save the file in the result folder described in the *Output files* section.

# Visualization

In the visualization section of the GUI you may see the feature extraction result, feature matching result and registration result after finishing the process of registration. In this section, after finishing the process you can choose to see the z-projection of overlapped stacks before registration (upper image), or z-projection of overlapped stacks after registration (lower image). You can use the toolboxes  for zooming in, zooming out, rotate or move the image to see the specific part of the image. To view this section, make sure you checked the *Preview* checkbox in section 4 of the GUI and choose proper Stack ID. The Stack ID you have choose will enable the visualization to visually present the result of registration before and after registration of the chosen stack with all its neighbors.

In this section, you may choose to see the layer view to scroll through the z planes of the registered stacks as shown in the below right image. You can either use the mouse scroll or the GUI scrollbar to go through the plains in Z. You are also able to see the current plane number along with the total number of planes.

Moreover, you may check the 3D view check box in section 6 of GUI to open the 3D view of the registered stacks as shown below.

# Parameters description

In addition to the parameters which are presented in section 3 of the GUI, there are several other parameters available in this software which are defined in the *parameters.m* file. These parameters are named based on their category as feature extraction parameters (*params.FE*), feature matching parameters (*params.FM*), global translation (*params.GT*), global affine (*params.GA*), and finally parameters of the blending named as params.BP. These parameters are categorized and listed in the following table along with their default value and description.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parameter | Section | Default | Description | Used In File (.m) |
| params.FE.smooth3BoxSize | Feature extraction | 3 | control the smooth3 filtering size | Find\_Seeds |
| params.FE.GaussianSize | 3 | control the Gaussian filtering size | Find\_Seeds |
| params.FE.IMboxSize | [5 5 3] | control the imbox filtering size | Find\_Seeds |
| params.FE.MLOG1 | 3 | control the Multi Scale LOG filtering size | Find\_Seeds |
| params.FE.MLOG2 | 3 | control the Multi Scale LOG filtering size | Find\_Seeds |
| params.FE.MLOG3 | 3 | control the Multi Scale LOG filtering size | Find\_Seeds |
| params.FE. windowsize | [4 4 4] | window to extract information around features | Find\_Seeds |
| paramsFMRansacMin | Feature matching | 15 | minimum matched points to run Ransac | Stitching\_3D\_Func |
| paramsFMDT | 50 | maximum distance of features to be consider in Hungarian Algorithm | Stitching\_3D\_Func |
| paramsFMMetric | 'ZMAD' | the similarity metric for feature matching | Stitching\_3D\_Func |
| paramsFMmaxiter | 1000 | maximum iteration of Ransac | Stitching\_3D\_Func |
| paramsFMdebug | 1 | debug mode | Stitching\_3D\_Func |
| params.GT.lamda | Global Traslation | 10^-6 | regularization of individual shifts | Global\_Optimal\_Translation |
| params.GT.eta | 1 | regularization of the overall shift | Global\_Optimal\_Translation |
| params.GT.lamda | Global Affine | 10^-6 | regularization of individual shifts | Global\_Optimal\_Affine |
| params.GT.eta | 1 | regularization of the overall shift | Global\_Optimal\_Affine |
| params.GA.nu | 1 | regularization of the overall deformation | Global\_Optimal\_Affine |
| params.GA.mu | 10^-1 | regularization of individual deformations | Global\_Optimal\_Affine |
| params.BP.extSize | Blending | [200 200 50] | size of the neighbor pixels for visualization | blending |
| params.BP.remove\_pad | 0 | Remove stack grey pad if exist | blending |
| params.BP.saveImages |  | 0 | Save result as tif file | blending |

# Steps to run the registration

To install run the registration process, follow these steps:

1. Install the software using install.exe file
2. After installation, open the software using *BrainStitcher.exe* file.
3. Click on Run button on GUI.

# About

## Contact information

The *BrainStitcher* is developed at the Neurogeometry lab at Northeastern University by Dr. Seyed Mostafa Mousavi Kahaki and Prof. Armen Stepanyants. All inquiries should be addressed to Prof. Armen Stepanyants at [a.stepanyants@neu.edu](mailto:a.stepanyants@neu.edu) or Dr. Seyed Mostafa Mousavi Kahaki at [kahaki@neu.edu](mailto:kahaki@neu.edu).

## Support

BrainStitcher is supported by the NIH under grant number NS091421.

## Related publications and posters

Seyed M. M. Kahaki, Shih-Luen W., and Armen Stepanyants, Automated registration of light microscopy stacks of images in space and time, Society for neuroscience, Washington DC, 2017.

## Contributions

Seyed Mostafa Mousavi Kahaki and Armen Stepanyants developed the software and prepared the documentations.