CATNIP Documentation

Systems Neuroscience Imaging Resource (SNIR)
National Institute of Mental Health, NIH

Written by

Snehashis Roy, Sarah Williams, Richard Edelmann, Ted Usdin

February 12, 2023

1 Introduction

CATNIP is whole brain **C**ellular **A**c**T**ivity estimation : a **N**ice **I**mage processing **P**rogram.

CATNIP was inspired by the ClearMap pipeline [1, 2]. CATNIP is designed to analyze cleared mouse brain images scanned in Lightsheet microscopes, e.g. iDISCO, CLARITY, and SHIELD type images. It can process very large (Terabyte scale) images. It was built for analysis of nuclear immunolabeling but can likely be adapted to other labels. There are three main steps of the pipeline,

1. Registration: The images are downsampled to approximately $25\mu m^3$ voxel size. They are corrected for any intensity inhomogeneity using N4 [3]. The inhomogeneity corrected and downsampled image is then registered to the Allen Brain Atlas (ABA) [4] via `antsRegistration` [5].

2. Segmentation : Since the labeled cells are generally of spherical shape on cleared images, they are segmented on the original image space using a fast radial symmetry transform (FRST) [6] . The transform provides a continuous valued "*membership*" type image, which can thresholded as various values to generate binary cell segmentation images.

3. Statistics and heatmaps: Cells in 3D are counted for each of the structural labels given by the ABA. Heatmaps of the cell counts are also generated for visualization and statistics.

The document is organized as follows. The installation process in described in Sec. 2. We provide two example datasets (one small and one large) to reproduce the pipeline results in Sec. 3. Then the atlas and acceptable input image types as well as various input arguments are described in Sec. 4 and Sec. 5. Finally details of various output images and the computed statistics are provided.

This work utilized the computational resources of the NIH HPC Biowulf cluster [1].

[1] https://hpc.nih.gov

2 Installation

CATNIP is currently configured to run on any 64-bit Linux workstation or cluster. For Windows, please use a virtualization software, such as VirtualBox or VMWare Player to install a 64-bit Linux distribution. We have tested CATNIP on Red Hat Enterprise Linux 7.9, CentOS 8, and Rocky Linux 9. We will release it for MacOS in the future.

CATNIP is primarily written in MATLAB, while parts (e.g., registration) of the pipeline are run via ANTs [5] toolbox. Note that it is **not** required to have an active MATLAB license as all codes are compiled. The pipeline is optimized to use minimum amount of memory at the cost of repeated read/write operations from disk. Minimum requirement to run the pipeline is a 8-core CPU, 32GB RAM, and a reasonably fast drive. A recommended requirement is 12-core CPU, 64GB RAM, and a fast SSD. With Terabyte scale images, 128GB RAM and an SSD with at least twice the size of the image are recommended to store temporary files. Administrator access is not required and not recommended for any of the installation process.

2.1 Install Dependencies

1. Download the code from Github to a suitable location,

   ```
   cd $HOME
   git clone https://github.com/snehashis-roy/CATNIP
   ```

2. Install MATLAB Compiler Runtime (MCR): Download the 64-bit Linux MCR installer for MATLAB R2019b (version 9.7) from here,

   https://www.mathworks.com/products/compiler/matlab-runtime.html

   Otherwise, download the installer from this direct link,

   https://ssd.mathworks.com/supportfiles/downloads/R2019b/Release/9/deployment_files/installer/complete/glnxa64/MATLAB_Runtime_R2019b_Update_9_glnxa64.zip

   Save the zip file in a suitable location, e.g. `/home/user/Downloads/`. Then extract the zip file, and run the installer within,

   ```
   cd ~/Downloads/
   unzip MATLAB_Runtime_R2019b_Update_9_glnxa64.zip
   ./install
   ```

   The installer will guide through the MCR installation process. Please install it in a suitable local directory, e.g. `/home/user/MCR`. Note that after installation, the final installed directory, called `MCRROOT`, will be `/home/user/MCR/v97`, where the `v97` denotes the version of the MCR. We will use this path in the next section.

3. Install ANTs: We have extensively tested and optimized CATNIP with ANTs v2.2.0. However, there is nothing special about this version that is specific to the pipeline. Feel free to use newer versions. Please let us know if newer versions do not work.

   - Basic users: Download the v2.2.0 of the ANTs binary executables from here, https://hpc.nih.gov/~NIMH_MHSNIR/CATNIP/ANTs-2.2.0-RHEL7.9.zip

It is currently only compiled for RHEL7.9. If you use different Linux, such as Ubuntu, please download the corresponding binaries from the Github repository[2]. After downloading, extract it to somewhere suitable, e.g. `/home/user/ANTs-2.2.0/`. Note that the required binary executables will then be located in `/home/user/ANTs-2.2.0/bin/`, which we will use in the next section.

- Advanced users: Download the source code of ANTs, unzip somewhere suitable, and build from source.

```
cd $HOME
wget https://github.com/ANTsX/ANTs/archive/refs/tags/v2.2.0.zip
unzip v2.2.0.zip && cd ANTS-2.2.0
mkdir build && cd build
make ../  -DCMAKE_INSTALL_PREFIX=/home/user/ANTs-2.2.0/install
make -j8
make install
```

After installation, the binaries will be located in `/home/user/ANTs-2.2.0/install/bin`.

2.2 Postinstallation Setup

1. Add ANTs binary path to shell's $PATH. The path is the **bin** folder that was set up in Sec. 2.1#3. Edit the bash shell's .bashrc (`gedit $HOME/.bashrc`) to add the following line,

   ```
   export PATH=/home/user/ANTs-2.2.0/install/bin:${PATH}
   ```

   For tcsh (`gedit $HOME/.bashrc`), add

   ```
   setenv PATH /home/user/ANTs-2.2.0/install/bin:${PATH}
   ```

   Then source the bashrc (`source $HOME/.bashrc`) or tcshrc (`source $HOME/.tcshrc`). If the ANTs binaries are successfully added to PATH, then simply running `antsRegistration` in a terminal should show a long list of its usage.

2. Add the MCR installation path, MCRROOT, obtained in Sec. 2.1#2. Open the following 15 shellscripts (.sh files) from the CATNIP folder with gedit,

   ```
   ApplyFRSTseg.sh
   ApplyFRST.sh
   create_heatmap.sh
   Downsample3D.sh
   fix_header.sh
   FlipImages.sh
   Generate_Stats.sh
   image_clamp.sh
   image_info.sh
   image_math.sh
   ```

[2] https://github.com/ANTsX/ANTs/releases

```
mask_correction_v2.sh
N4Process.sh
nii2tiff.sh
remove_background_noise.sh
upsample_image.sh
```

In each of them, replace the line containing `MCRROOT=/usr/local/matlab-compiler/v97` to the following,

```
MCRROOT=/home/user/MCR/v97
```

where the desired MCRROOT path was obtained in Sec. 2.1#2

3 Reproducible Dataset

We provide two datasets to reproduce the pipeline. The first dataset contains a iDISCO cleared hemisphere of a mouse brain. The size of the image is $2560 \times 2160 \times 1030$ voxels, with $3.77 \times 3.77 \times 5\mu\text{m}^3$ voxel size. An example artifact mask is also provided with the dataset. Please see Sec. 5 for the explanation of artifact masks. The image along with the artifact mask(15GB) and the processed results (14GB) can be downloaded from here.
https://hpc.nih.gov/~NIMH_MHSNIR/CATNIP/example_data.zip
https://hpc.nih.gov/~NIMH_MHSNIR/CATNIP/example_data_processed.zip

The image was processed with the following command,

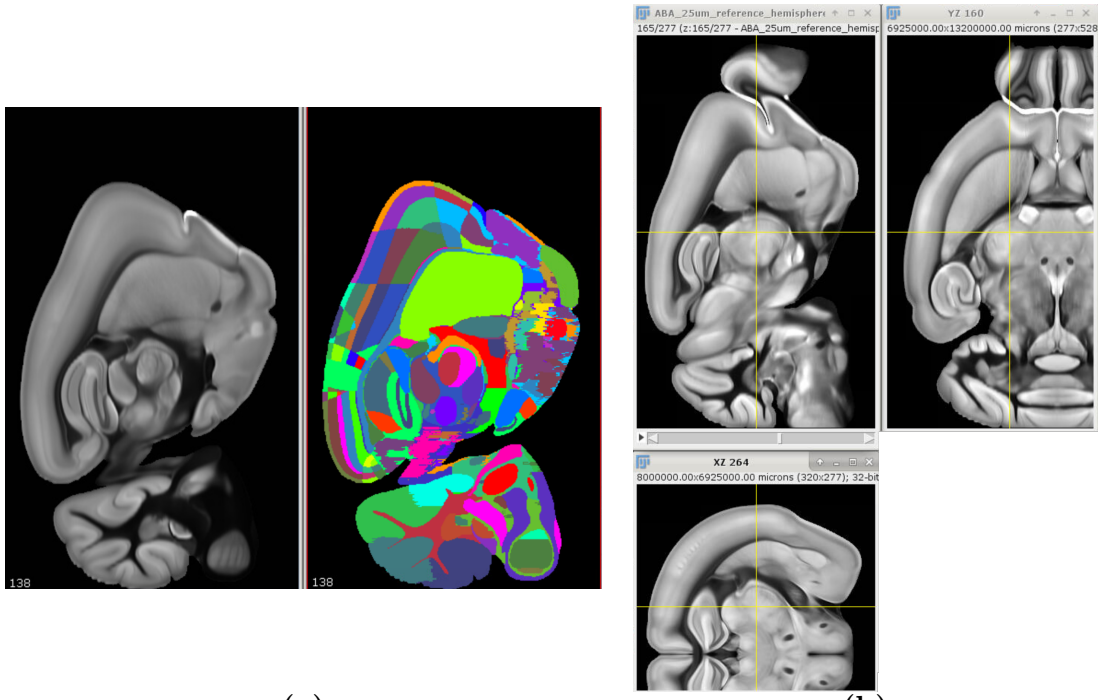
```
./CATNIP.sh --c640 example_data/ --o ./ --ob no --lrflip yes --udflip yes \
--thr 1000:1000:5000 --dsfactor 6x6x5 --cellradii 2,3,4 --ncpu 16  \
--exclude_mask example_data_artifact_mask.tiff --atlasversion v2
```

The second dataset contains a CLARITY cleared images of dimension $13413 \times 10130 \times 1786$ voxels, $0.94 \times 0.94 \times 4\mu\text{m}^3$ voxel size. The original image (unzipped 453GB) and the processed results (323GB) can be downloaded from here,
https://hpc.nih.gov/~NIMH_MHSNIR/CATNIP/example_data_large.zip
https://hpc.nih.gov/~NIMH_MHSNIR/CATNIP/example_data_large_processed.zip

The image was processed with the following command,

```
./CATNIP.sh --c640 example_data_large/ --o ./ --ob no --udflip no --lrflip no \
--thr 20000:10000:100000 --dsfactor 25x25x6 --cellradii 3,4,5,6,7,8 --ncpu 8  \
--atlasversion v2
```

Explanations of the command line arguments are given in Sec. 5. The output results are also explained in Sec. 6.



(a) **(b)**

Figure 1: Atlas orientation: **(a)** ABA atlas downloaded from ClearMap resources (https://idisco.info/clearmap-2/) after reorientation to sagittal view along with the discreet label image. **(b)** An orthogonal view from Fiji showing that the atlas has one hemisphere and some part of the other hemisphere to account for proper registration of midline labels.

4 Atlas and Input Image

All images in this document are shown from Fiji. All definitions of orientation are also given w.r.t. Fiji. The input to the pipeline should be a folder with multiple 2D tif images in the atlas orientation corresponding to the 640nm channel. We acquired images one hemisphere at a time in the sagittal orientation. Therefore, the ABA atlases (downloaded from ClearMap resources[3]) were also rotated to the sagittal orientation. Fig. 1(a) shows our rotated sagittal view of ABA atlas and the corresponding label image. For better registration and to capture all midline structures, the imaging should go beyond the midline and into the other hemisphere. The atlas and the label image therefore are padded with 49 additional slices further into the other hemisphere from the midline slice. Fig. 1(b) shows an orthogonal view from Fiji describing the extent of the atlas having more than exactly one hemisphere.

In the original ABA atlas, there were a few labels that had arbitrarily high values, causing problems in most visualization softwares. So we updated their label values to smaller numbers to keep the range of the image to 16bit unsigned integer range. They were also unlabeled in the label information file. In our atlas label information, we have assigned them the correct labels by manually checking the website [4]. The corrected labels are shown in Tab. 1. Note that the label information file and the label images still contain two labels 512 (Cerebellum) and 12009, which we believe are either incorrect or unlabeled. It is preferred to exclude these two from all analysis.

The input image should follow these three requirements to match to the atlas,

[3] https://idisco.info/clearmap-2/
[4] https://mouse.brain-map.org/

Label Value	Label Name	Label Acronym
12001	Primary somatosensory area unassigned layers 1-3	SSp-un 1-3
12002	Primary somatosensory area unassigned layers 4-6	SSp-un 4-6
12003	Anterior area layers 1-6	VISa 1-6
12004	Rostrolateral visual area layers 1-6	VISrl 1-6
12005	Laterointermediate area layers 1-6	VISli 1-6
12006	Postrhinal area layers 1-6	VISpor 1-6
12007	Frontal pole layers 5-6a	FRP 5-6a
12008	Frontal Pole layer 5-6b	FRP 6b

Table 1: For visualization purpose, we modified some atlas label values from the ABA atlas information list, where the labels had arbitrarily high value.

1. Input image must be of one hemisphere in sagittal orientation (*not* the whole brain), but can be left-right or up-down flipped.

2. The depth (z-direction) must be toward the midline.

3. The image should contain some of the other hemisphere.

Some examples of orthogonal views of unacceptable images are shown in Fig. 2 top row. It is

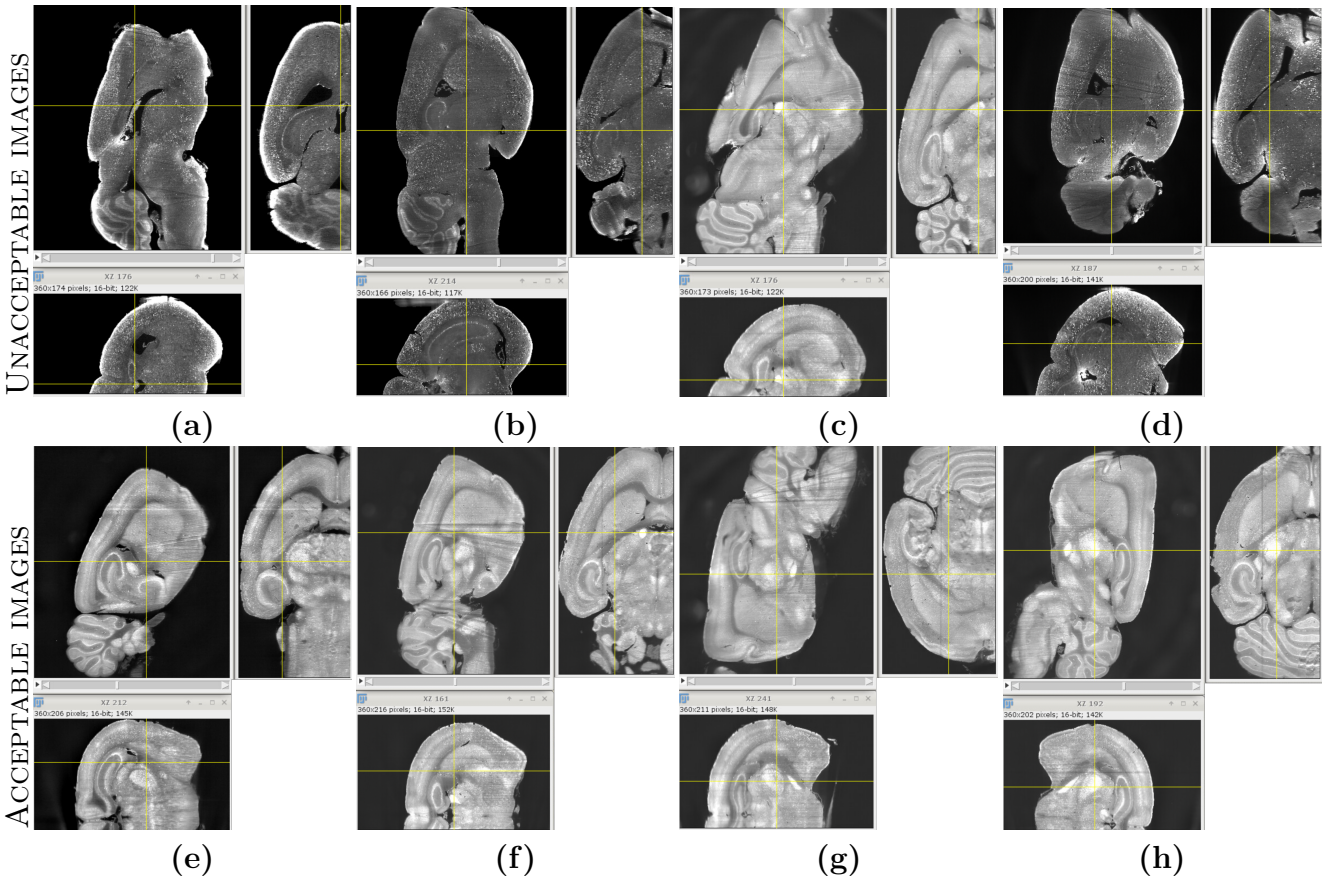


Figure 2: Examples of unacceptable (top row) and acceptable (bottom row) input images. Up-down or left-right flipped images are acceptable.

strongly recommended to check for the conformity of the input image to the atlas via orthogonal views in Fiji before processing through the pipeline. Note that the unacceptable hemispheres do not either contain midline structures (**(a)**-**(c)**) or sufficient amount of the other hemisphere (**(d)**). Even if the experimental setup does not consider midline structures or cerebellum, it is imperative that they are included in the image for the 3D registration. Since the olfactory bulbs are frequently missing from the images, we have created a second copy of the atlas by removing the olfactory bulb labels.

5 Input Arguments

In this section, we describe various mandatory and optional input argument to the main script `CATNIP.sh`. The user should first visualize the image (preferably in Fiji) to identify various input arguments as listed. An example command line usage of the pipeline is as follows,

```
./CATNIP.sh --c640  /home/user/input/ --o /home/user/output --ob yes --lrflip yes \
--udflip yes --thr 1000:1000:5000 --dsfactor 6x6x5 --cellradii 2,3,4 --ncpu 12  \
--exclude_mask  /home/user/example_data_artifact_mask.tiff --atlasversion v2 \
--bg_noise_param 50,1.05 --mask_ovl_ratio 0.33
```

5.1 Mandatory Arguments

1. `--c640` : This is the input image, formatted as 2D tifs in a single folder. Note that the filenames of the individual tif images must be sorted alphabetically as a z-stack. E.g., filenames such as `ABCD_1.tif`, `ABCD_2.tif`, .. , `ABCD_100.tif` are not allowed because they are not alphabetically sorted. In that case, filenames should be `ABCD_00001.tif`, `ABCD_00002.tif`, .. , `ABCD_01000.tif` etc. The orientation of the input image is described in Sec. 4.

   Usage: `--c640 /home/user/input_image/`

2. `--o` : This is the output directory where results will be written. A subfolder will be created with `basename` of the input image and a timestamp for easier identification in case of multiple runs.

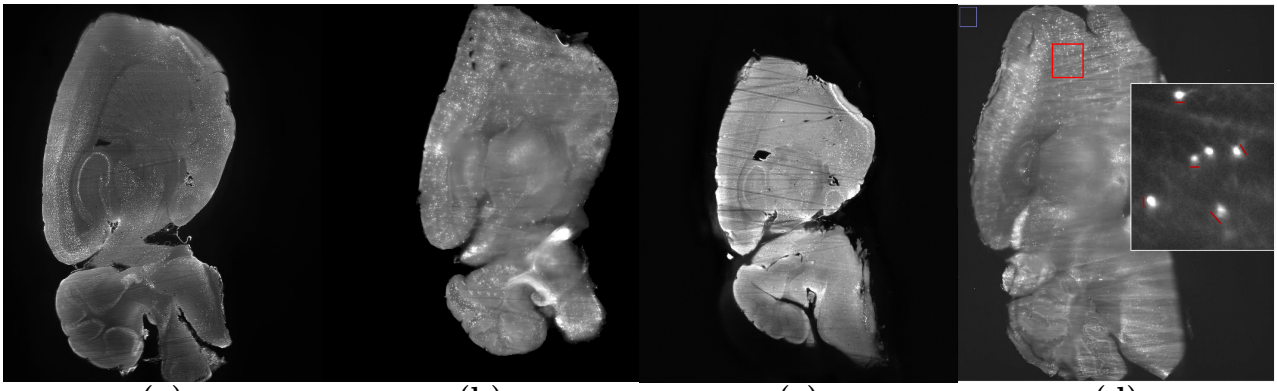
   Usage: `--o  /home/user/output_folder/`

3. `--ob` : This is a yes or no flag indicating if the image contains olfactory bulb. If the olfactory bulb is partially present in the image, then the 3D registration can be inaccurate in that region. In that case, it is recommended to process the image with both flags separately, check the registrations manually, and choose whichever flag gives better registration. The registration output is listed in Sec. 6.

   Usage: `--ob yes`

4. `--lrflip` : This is a yes or no flag indicating if the image is left-right flipped w.r.t the atlas orientation. See Sec. 4 and Fig. 1 to check the correct atlas orientation. Example of a flipped image is shown in Fig. 2(h).

   Usage: `--lrflip yes`



(a) **(b)** **(c)** **(d)**

Figure 3: Example of various image sizes and cell radii are shown. **(a)** An adult mouse brain cleared with iDISCO, image size $2560 \times 2160 \times 1030$ voxels, a downsampling factor (`--dsfactor`) of `6x6x5` was used. **(b)** A SHIELD cleared brain with significant enlargement, image size $4796 \times 4038 \times 1538$ voxels, a downsampling factor `9x9x7` was used. **(c)** A slightly smaller size cohort cleared with iDISCO, image size $2160 \times 2560 \times 994$ voxels, downsampling factor `5x5x4` was used. All of the images have the same voxel size $3.77 \times 3.77 \times 5 \mu m^3$. **(d)** An example of the range of cell diameters (red lines) are shown. A zoomed in view of the red box is given inset. The range of radii is provided via `--cellradii` argument. See Sec. 5 for details about the arguments.

5. `--udflip` : This is a yes or no flag indicating if the image is up-down flipped w.r.t the atlas orientation. Examples of a flipped image is shown in Fig. 2(g).

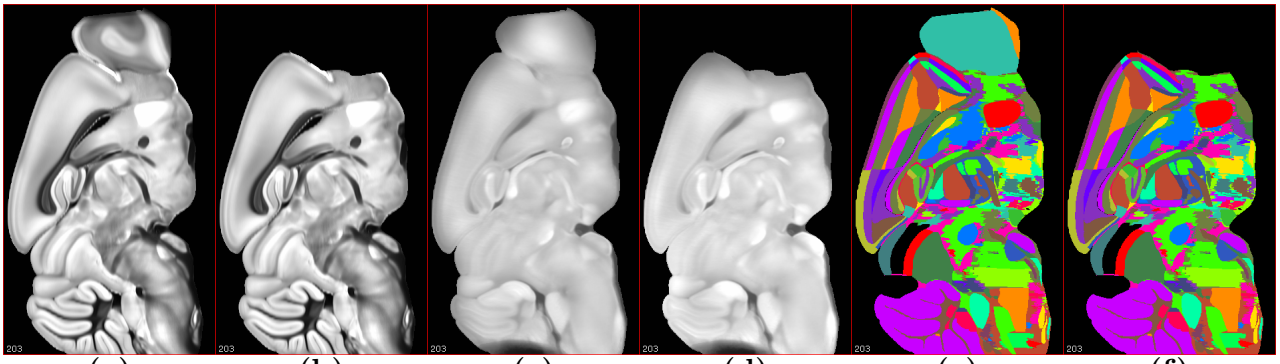
   Example: `--lrflip yes`

6. `--dsfactor` : This is a downsampling factor to downsample the image into the ABA space for 3D registration. The ABA is an adult mouse brain atlas with $25\mu m^3$ voxel size. So a rule of thumb of choosing the downsampling factor is to downsample the image into approximately $25\mu m^3$ isotropic voxel size. E.g., an appropriate downsampling factor for a $3.77 \times 3.77 \times 5$um image is $6 \times 6 \times 5$. Similarly, a $1 \times 1 \times 4\mu m^3$ image needs a $25 \times 25 \times 6$ downsampling to make it close to $25\mu m^3$ isotropic.

   However, this downsampling factor is flexible and not strictly tied to the voxel size. There might be cases where the overall size of the brain is inherently smaller than an adult mouse brain, e.g., juvenile mouse brains. If it is estimated that the average brain dimension of the experimental cohort is generally 20% smaller than an adult mouse brain at $3.77 \times 3.77 \times 5$um, then a smaller downsampling factor should be used, e.g. $5 \times 5 \times 4$. On the other hand, SHIELD or CUBIC clearing can inflate the overall size of the brain compared to iDISCO. See Fig. 3(a)-(c) for some examples of brain size variation. If it is estimated that the size increases by 50%, a larger downsampling factor such as $9 \times 9 \times 7$ can be used instead of recommended $6 \times 6 \times 5$ at $3.77 \times 3.77 \times 5$um. If the image resolution is non-standard, it is recommended to check the atlas registration, as described in Sec. 7, and experiment with few other downsampling factors according to the voxel size as well as the overall size of the brains comparing with an adult mouse brain.

   Usage: `--dsfactor 6x6x5`

7. `--cellradii` : This is a range of radii (comma separated) in pixels observed in the image. An example of choosing the range is shown in Fig. 3(d). It is recommended to look at various



(a) (b) (c) (d) (e) (f)

Figure 4: We use two versions of ABA atlas, **(a)-(b)** original one downloaded from ClearMap resources, and **(c)-(d)** a blurred version with less detail (NEED MORE INFO). They have 8 modified labels, as described in Sec. 4 and Table 1.

regions of the brain and estimate a range of cell radii. The default value is `2,3,4`, indicating spherical blobs of diameters 4 to 8 pixels are to be chosen as a potential cell. It is used to compute the potential cell boundary pixels, as described in [6] (see Fig. 3 of the paper for a visual interpretation).

Usage: `--cellradii 2,3,4,5,6,7`

5.2 Optional Arguments

1. `--ncpu` : This is number of parallel processes to use. Usually, the amount of required memory is almost directly proportional to the number of parallel processes. For small images (e.g., $2560 \times 2160 \times 1500$ pixels, $3.7 \times 3.7 \times 5\mu\text{m}^3$), 12 or 16 is a reasonable number. For very large images (e.g., $12000 \times 8500 \times 3000$ pixels, $0.95 \times 0.95 \times 2.5\mu\text{m}^3$), use 8.

   Usage: `--ncpu 12`

2. `--thr` : This is a Matlab style string (start:increment:stop) indicating values to threshold the FRST (see Sec. 6 for details) images to generate binary cell segmentations. Since FRST provides an arbitrary valued membership type images, the thresholds are arbitrary and image specific. It is recommended that a range of thresholds are used, so that segmentations for each of the thresholds are generated. Example, $10000 : 1000 : 15000$ means 6 segmentation images will be generated at thresholds $10000, 11000, \ldots, 15000$. If not mentioned, default is $45000 : 5000 : 60000$. At lower threshold, oversegmentation may occur and multiple cells may be combined together. At higher thresholds, low intensity cells may not be detected.

   It is recommended that one image from a cohort is processed with default values of the threshold, then the user should look at the FRST images and decide a range of thresholds to use on all images in a cohort.

   Usage: `--thr 1000:1000:5000`

3. `--atlasversion` : We provide two versions of the atlas. The first one (denoted by `v2`) is the same as the ClearMap resources[5], also used in ClearMap2[6] pipeline. The second

[5] https://idisco.info/clearmap-2/
[6] https://github.com/ChristophKirst/ClearMap2

one (denoted by `v1`) is a blurred version. Both of them contain a corresponding alternate version without the olfactory bulb (OB). The raw images without OB were created by simply deleting pixels corresponding to the various OB regions obtained from the label image. The atlases are shown in Fig. 4.

On most of the images, any one of the two atlas sets work fine for 3D registration with ANTs. Only for a very few cases, we have observed that only one of the atlases provide sufficiently adequate registration. We hypothesize that this is caused by the intensity distribution mismatch between the atlas [7] and 640nm Lightsheet images. As the 640nm Lightsheet images do not show any structural information, the blurred version of the atlas (`v1`) can sometime register better. This is also caused by our choice of mutual information as the similarity metric in the ANTs registration. Empirically, we have found that mutual information is more robust that the alternative, normalized cross correlation.

Usage: `--atlasversion v2`

4. `--exclude_mask` : Sometimes the input image can contain various artifacts, such as shadows originating from impurities in the medium or air bubbles. In case of imaging artifacts, the cell segmentation may become unreliable. If part of a region contains artifacts, it is preferred that the cell counts from that region are excluded from any analysis. The exclusion mask is a way to exclude regions which has significant artifact. It is a binary mask indicating bad regions or artifacts in the image.

   This mask must already be in the "atlas orientation", i.e. sagittal and cerebellum in bottom left side. See Fig. 1(a) for the definition of the atlas orientation. This image must be in one of the following space,

   - It can have the same dimension as the original image "after" orienting to atlas orientation. In this case, the image must be TIF (.tif or .tiff).
   - It can have the same dimension as the downsampled image. In this case, the mask must be a NIFTI (.nii or .nii.gz) image. Since it is easier to draw on a small image, it is recommended to run the pipeline once with correct downsampling factor to generate all outputs, then draw a binary mask on the `640_downnsampled_AxBxC_brain.nii.gz` image.

   Note that this mask will *not* be reoriented based on the flip flags. We recommend MIPAV or ITKSnap to draw mask on NIFTI images.

   Usage: `--exclude_mask /home/user/mask_in_downsampled_space.nii.gz`

5. `--mask_ovl_ratio` : This is number between 0 and 1. It is used only when an exclusion mask is provided. A ratio of 0.5 means a label is going to be ignored if its overlap volume with the mask is less than half of its total volume. Usually, a small overlap with the exclusion mask may be ignored because the manual artifact delineation may just overlap with a big region by a few pixels. Default is 0.25, meaning a label must have at least one quarter of its volume identified as artifacts in the exclusion mask to be ignored. The cell counts of such labels are given by -1.

   Usage: `--mask_ovl_ratio 0.33`

6. `--bg_noise_param` : Before atlas registration, the background noise from the downsampled image is removed for better atlas registration with ANTs. This is an optional comma separated pair, e.g. $50, 1.05$, where 50 indicates the percentile of the image histogram that is treated as an initial background noise threshold. Then this threshold is iteratively increased by 5% (hence 1.05) until the foreground does not change any more. For images with heavy noise, use higher percentiles and larger stepsize, such as $60, 1.1$. For images without much background noise, use lower numbers, such as $40, 1.05$. Note that if the physical image dimension is quite larger than actual brain dimension (e.g. Fig. 3(c), it may be treated as higher background noise, as opposed to Fig. 3(a) where image dimension is very similar to hemisphere dimension.

   Usage: `--bg_noise_param 50,1.05`

6 Output

In this section, various outputs of the pipeline are described.

1. `640_downsampled_AxBxC.nii.gz` : It is the downsampled version of the input image in atlas space.
2. `640_downsampled_AxBxC_brainmask.nii.gz` : It is the mask created after removing the background noise.
3. `640_downsampled_AxBxC_brain.nii.gz` : It is the downsampled input image after background removal. The atlas is registered to this image.
4. `hemispheremask_AxBxC.tif` : It is a binary mask denoting the hemisphere. All cell count statistics are only computed on one hemisphere.
5. `640_N4_masked` : The brainmask is upsampled to the original image space and so is the hemisphere mask. Then the original image is masked by them to contain the single hemisphere.
6. `atlasimage_reg.nii.gz` : It is the registered atlas image in the downsampled space.
7. `atlaslabel_def.nii.gz` : It is the deformed atlas label volume.
8. `atlaslabel_def_origspace_masked` : The deformed atlas label volume is upsampled to the original image space and masked by the brain mask and hemisphere mask.
9. `640_FRST` : A fast radial symmetry transform is applied on the masked image in original space. It is a continuous valued "membership" type image with 32-bit floating point values.
10. `640_FRST_seg` : This folder contains binary cell segmentation computed at the assigned thresholds as well as csv files containing cell counts and label volumes (in pixels) for each of the labels given in the *atlas_info.txt* file in the atlas directory. Please refer to Table 1 for information on 8 modified label numbers that we changed from the original ClearMap2 label information, as the original numbers were too large to fit in UINT16 range. See Sec. 4 for details.
11. heatmaps_atlasspace : This folder contains cell density heatmaps, similarly computed as the ClearMap pipeline. In addition to region based cell count statistics, these heatmaps can be used to compute voxel based statistics.

7 QA

References

[1] C. Kirst, S. Skriabine, A. Vieites-Prado, T. Topilko, P. Bertin, G. Gerschenfeld, F. Verny, P. Topilko, N. Michalski, M. Tessier-Lavigne, and N. Renier. Mapping the fine-scale organization and plasticity of the brain vasculature. *Cell*, 180(4):780–795, 2020.

[2] N. Renier *et. al.* Mapping of brain activity by automated volume analysis of immediate early genes. *Cell*, 165(7):1789–1802, 2016.

[3] N. J. Tustison, B. B. Avants, P. A. Cook, Y. Zheng, A. Egan, P. A. Yushkevich, and J. C. Gee. N4ITK: Improved N3 Bias Correction. *IEEE Trans. Med. Imaging*, 29(6):1310–1320, 2010.

[4] Q. Wang and S.-L. Ding *et. al.* The Allen Mouse Brain Common Coordinate Framework: A 3D Reference Atlas. *Cell*, 181(4):936–953, 2020.

[5] B. B. Avants, C. L. Epstein, M. Grossman, and J. C. Gee. Symmetric diffeomorphic image registration with cross-correlation: evaluating automated labeling of elderly and neurodegenerative brain. *Medical Image Analysis*, 12(1):26–41, 2008.

[6] G. Loy and A. Zelinsky. Fast radial symmetry for detecting points of interest. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(8):959–973, 2003.

[7] S. W. Oh and *et. al.* A mesoscale connectome of the mouse brain. *Nature*, 508(7495):207–214, 2014.