CTB Tracing Program: Joe Holden

This program operates using the methods outlined in Crish et al., 2010 in PNAS:

*“Background intensity for each brain section was set independently for normalization using pixel strength of the nonretinorecipient SC (layers IV–VII) and the periaqueductal gray. Using layer IV as the ventral border, we outlined in each section the boundaries of the superficial SC, which was partitioned into* ***6-μm bins from medial to lateral****.* ***For each bin, the area of pixels with CTB signal above background was divided by total pixel area to determine CTB density****. This was assigned a colorimetric representation ranging from 0% (blue) to 100% (red) at each mediolateral location in the SC section. Using section thickness and intersection distance, we adjoined sections to construct a colorimetric representation of CTB density across the retinotopic SC map.* ***For each SC, we determined the fraction of intact retinotopic map, defined as the percent area with CTB signal ≥70% density.****”*

Assuming the same background intensities are chosen (be aware of possible conversions between 8- and 16-bit files) and the same outlines are traced, my program will give the exact same answer as the Image Pro macro. The algorithm is the same- the coding language is different.

This program is run in two distinct “phases”. The first requires Fiji ImageJ and the second requires Python. Fiji can be downloaded from the internet easily, as can Python. However, there is a bit more to installing Python than Fiji. This will only need to be done once, and many tutorials exist online. In brief, go here: <https://www.python.org/downloads/> and use the Windows .exe installer. I have not verified the code runs “nicely” on Mac and there are differences in path naming that we don’t want to deal with for this. If you know how to code and want to use a Mac, go ahead and edit it yourself. In the installation, make sure to check the box that says “Add Python 3.X to the Path”. It also helps if you install an IDE (integrated developer environment). This is a fancy text editor that can run your programs in a nice environment. Use PyCharm Community (free) edition. <https://www.jetbrains.com/pycharm/download/?section=windows> . Add it to the Path as well during the installation so PyCharm can “find” your Python install.

Steps

1. Image your SC brains. Do this at 4X resolution on the Nikon fluorescent microscope. When you take your images, make sure both the Left and Right SC are visible in the same image. This is important because Analysis of the Left and Right SC will occur at the same time.
   1. Images are saved as .nd2
      1. If you do not use the Nikon scope, you will have some editing to do. The Fiji macro will name the ROIs we make based on the .nd2 extension. This is important in line 150 of ctb\_trace.py.
   2. When you save images you need to follow a consistent naming convention. The overall algorithm works by breaking up the SC into vertical strips, seeing how many pixels are above background threshold in that strip and assigning a density value to that strip. The is done for all sections in the anterior-posterior axis, and we need to keep track of the order of the sections.
      1. The file regex\_sort.py contains the functions to sort your filenames and determine the order they are in. Do not edit the file unless you have a strong reason to use a different naming convention. You will have to edit the file using regular expressions if you use a different method than described below.
      2. Files should be named like the following example: “AB41 slide 2 slice 12\_002.nd2”. The critical parts here are highlighted. When sectioning the brain there is usually two slides. If not don’t worry because all the files can be named like ‘slide 1 slice X’. The program sorts these names first by the number that follows ‘slide’ and then by the number that follows ‘slice’. **The space between the word slide and the number as well as the space between slice and the number are critical. It must match exactly so don’t use underscores or something.**
2. After you have the ND2 images, you need to get outlines of the SC. We do this in Fiji.
   1. Open Fiji and load the macro ‘Save ROI.ijm’
   2. This macro will add your ROIs to the ROI manager and rename them so we can keep track of which ROIs came from which image.
   3. The macro assumes you will trace the Left ROI first and it assumes you have a Left and Right SC to trace. If you do not, you will have to rename these ROIs manually or make your own macro to account for this.
   4. Use the polygon ROI tool. Make sure to close the ROI off.
      1. Once you have closed the ROI off, hit RUN on the macro.
      2. See the picture below which highlights the polygon tool, an example trace, where the macro ‘RUN’ button is and how the rois are renamed in the manager.
      3. After you hit RUN and it adds the ROI for the Left side, click off the ROI and trace the Right side. Hit RUN.
      4. Notice the macro would name the files incorrectly if you only had a Left SC to trace and not a Right one or vice versa. You can write a fancier macro but all the brains I’ve imaged have both sides so there’s not much of a need; this is a simple solution.
   5. After you trace the SCs for one image, close the image and open the next image. Trace it in the same way.
   6. Once you have finished tracing all images, select all the ROIs in the manager (Ctrl A) and hit the Save button. It might be under the ‘More’ button depending on your version.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

1. Once you have traced all the SC, we move to the phase which uses Python.
   1. In PyCharm, open the file “DIRECTORY\_SETUP.py”
      1. Edit line 5 to reflect your project name. This is not super important, its just going to create a directory tree where we will put all our ROI files we created and also save 8-bit Tif versions of the ND2 files. All of this is done automatically.
      2. You also must edit line 8. This line defines the variable ‘animals’ which is a list of animal numbers.
         1. This line should look something like: animals = [3418, 3419, 3420, 3421]
         2. You must give animal numbers here. If your experiment didn’t use animal numbers just make them up at this point.
      3. Edit line 11. This is your ‘working directory’. You need to write a path to a directory where you want everything to save.
         1. A proper path is in quotes and folders are delineated with a forward slash.
      4. Go ahead and run this file
         1. A folder is not generate in your working directory with the name of your project. Sub folders are created for each animal for ROIs and TIFFs.
         2. Running the file also generates a color bar PNG which will be useful later.
      5. Copy all the ROIs for each animal into their folders now. In the image below, my project name is ‘Test Project’ and I have copied the ROIs for animal 40 into the folder ‘40 roi’ which was generated automatically.

A screenshot of a computer

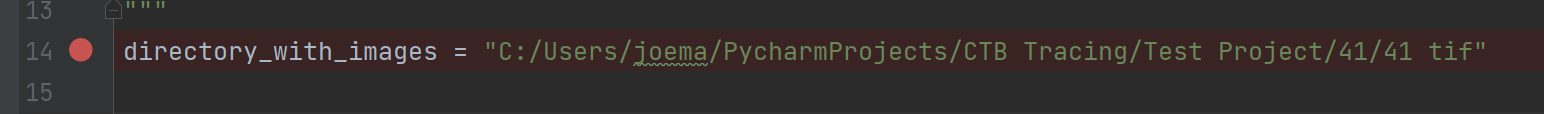
Description automatically generated

* 1. Let’s move back to ImageJ for a minute. We need to generate 8-bit TIFF files for each ND2. There’s no point doing this in the Nikon software because it’s slow as shit. Just do it automatically in Fiji using the macro ‘Batch ND2\_TIF8.ijm’.
     1. When you open the macro, edit the first line ‘save\_dir’. This must be a path name in double quotes, with folders delineated by forward slashes, and ends in a forward slash. This path is to the specific TIFF folder (that we automatically generated) that is in your project directory.
     2. When you hit RUN, a window pops up and you have to direct it to where the ND2 files are saved for that animal.
     3. Do this for all animals. When you finish, each animal folder in the project directory should have TIFFs and ROIs.

A screenshot of a computer

Description automatically generated

1. Back to Python. We are going to get the background intensities for each image now.
   1. Open the script “get\_thresholds.py”
   2. Edit line 14. This should be a path to a TIFF folder for one of the animals.



* 1. Run the script
  2. A window will pop up with a brain and some of the SC colored in. There is a slider bar up top.

A screenshot of a cell phone

Description automatically generated

* 1. Adjust the slider until the Red color covers the SC. This part is very important. If you have too much red, you are telling the program that transport is 100%. What I’d recommend is starting with the slider way to the right and see with just the fluorescent image if there are any big deficits. You will want these reflected. Slowly move the slider over until it paints in the region you are confident is SC. This is a subjective process so just be consistent however you do it.
  2. You will adjust the background for the left and right side independently. Adjust the left side first. Once you are happy with it, tap the ‘Q’ key on your keyboard. This locks in the background value. Adjust the slider now for the right side. When you’re happy with it, tap ‘W’. This locks in a value and pulls up the next image. Repeat until you have processed all the images for that animal. It only takes a few minutes.
  3. Repeat this process for all the animals.

1. All the manual work is done. Open ‘main.py’ and hit run.
   1. Heatmaps are generated for both SC of each animal and a master excel sheet is generated with all the transport percentages.
   2. The heatmap folder is in the project directory, as is the CTB transport excel sheet. It is timestamped.
   3. The ‘Plasm Colorbar.png’ is an image showing you the heatmap colors mapped to a range of 0-1 (0 – 100% transport). You will need this for a publication.

A screenshot of a computer

Description automatically generated