**Complete Guide to 2P Code Installation and Use**

This toolbox is designed to allow the user to present visual stimuli using pyschophysics toolbox and sync experimental events with Bruker Prairie Two Photon Systems. This toolbox also contains a number of analysis scripts to extract stimulus driven Calcium transient activity from imaging recordings.

**Hardware requirements:**

1. One Bruker-Prairie two photon imaging system (Ultima etc)
2. One Windows computer with graphics card compatible with psychtoolbox
3. RECOMMENDED, separate Windows analysis computer with good GPU (GTX 1070 and above).
4. Interface box to communicate between stimulus computer and Bruker system (Measurement Computing USB-1408FS preferred)

**Software requirements for analysis computer:**

1. Download toolbox and add to Matlab path
2. Install psychophysics toolbox (<http://psychtoolbox.org/>)
3. If you want to be able to use no rigid motion correction please clone the non-rigid motion correction toolkit (<https://github.com/flatironinstitute/NoRMCorre>)
4. Install an up to date version of FIJI (<https://fiji.sc/>)
5. Connect FIJI with matlab as explained here (http://bigwww.epfl.ch/sage/soft/mij/) NB, instead of using ij.jar, place the up to date version from your FIJI package (FIJI.app/jars), it will be named something like ij-1.52g.jar into the MATLAB folder.
6. Install Cell Magic Wand into FIJI (<https://www.maxplanckflorida.org/fitzpatricklab/software/cellMagicWand/>) OR <https://github.com/GrimmSnark/Cell_Magic_Wand>)
7. Install CaImAn-MATLAB analysis package from github, this package uses some of their functions (<https://github.com/flatironinstitute/CaImAn-MATLAB> ).
8. You may need to increase your java heap size for FIJI and matlab to work with large images see (https://www.mathworks.com/matlabcentral/answers/92813-how-do-i-increase-the-heap-space-for-the-java-vm-in-matlab-6-0-r12-and-later-versions) NB use the java.opts method.
9. You will need to modify the "*intializeMIJ.m*" to your local FIJI path.

**Software requirements for stimulus computer:**

1. Download toolbox and add to Matlab path
2. Install psychophysics toolbox (<http://psychtoolbox.org/>)
3. Install Measurement Computing USB-1408FS and MC package for Matlab

**Stimulus computer setup:**

This system uses an analogue to digital conversion to communicate between the stimulus computer and the Bruker-Prairie two photon imaging system. This is achieved by splitting up the output range of the MC USB 1408FS (0-4V) into 255 discrete levels which are used to produce pulses by the stimulus computer which are recorded by the Bruker-Prairie two photon imaging system. These pulses are used event numbers to synchronize the experimental stimulus to the functional imaging. This conversion requires set up in the following procedure:

1. Follow MC USB1408FS instructions to connect the first analogue channel out (AO0) to the first analogue in (AI0) on the Bruker- Prairie System
2. Follow MC USB1408FS instructions to connect the digital port A to the trigger input on the Bruker- Prairie System
3. Set up TSeries run on the Bruker- Prairie System, ensure that the ‘start with external trigger button’ is selected. Be sure that you have the ‘voltage recording’ set up to record the first analogue channel.
4. Run *testDAQOutSignal* and save the resulting TSeries data. We are only interested in the voltage excel file.
5. Copy this TSeries folder to the stimulus computer and run *readEventFileSetup.m* to create *PrairieVoltageInfo.mat*, which contains the event voltage level information which is the basis of the stimulus computer to Bruker computer communication.
6. Open *readEventFilePrairie* and change the “keyFilepath” variable to the location of the newly created *PrairieVoltageInfo.mat.*
7. Set up stimulus monitor position and create entry in *degreeVisualAngle2Pixels* so that stimulus are created as the correct size. NB Make sure all PTB experiment scripts run use the correct setup number.

Now you can run stimulus experiment code from the PTB\_Experiments folder.

**Running an Experiment**

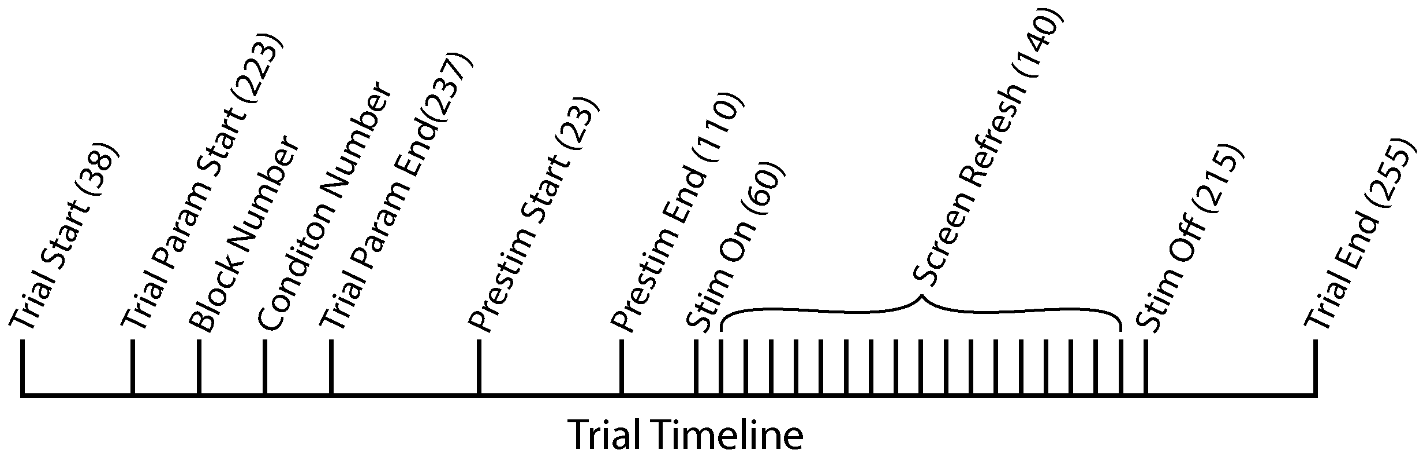
Please refer to the Two Photon Rig Visual Stimulation Experiment Master Protocol document.

**Experiment Trial and Information Structure**

All stimulus parameters for a given experiment are saved in a stimParams\_DATE.mat file on the stimulus computer. This is then copied over to the experimentStructure.mat for each imaging experiment during *prepData*.

As stated above, event communication between the stimulus computer and the Bruker computer is achieved through analogue to digital conversion. Calibrated amplitude voltage pulses are used to denote 8bit (256) event numbers. These event numbers and their experiment value (stim on, stim off etc) are held in *prairieCodes*. There is a lot of room if you need to add more events, be sure to add them to *prairieCodes* in both the stim **and** analysis computers.

Basic trial structure is show below:



General timeline of trial events: Numbers in parenthesis are the numeric events used for each code.

**Data Processing Pipeline**

Preprocessing

Run *prepData*, *prepDataBatch* or *prepDataBatchWithOrientationPixelPref* depending on your specific needs. This will run motion correction and create average images for ROI selection and create the *experimentStructure.mat* for each imaging run.

Cell ROI selection and calcium trace extraction

*runCaAnalysisWrapper* runs semi-automated analysis but needs manual curation for Cell ROIs. This produces calcium traces for each selected cell and splits them out into each condition.

In Matlab type ‘help experimentStructureClass’, ‘help experimentStructureClass.fieldname’ to get more details on how each field is calculated. The same can be done with any loaded experimentStructure by called ‘help experimentStructure’

Analysis

Cell Based Analysis

*checkCellROICOContourOverlap* compares cell ROI locations from functional imaging to a cytochrome oxidase boundary map. This gives you a cell identity for CO interpatch or patch.

*checkDualChannelExpression/\_wrapper* checks whether cell ROIs are visible in both channels of recording. For example, does a cell show up in functional channel and structural channel.

*calculateOSIPopulation/\_wrapper* calculates the orientation selectivity indices and metrics for each cell. See function help for more details on what is calculated.

*convertPixelOrientationSelectivityToCell* calculates and plots cell ROI average orientation preference and selectivity maps based on the pixel selectivity maps produced by *pixelwiseOrientationSelectivity.*

*gatherPopulationOSI* collates orientation selectivity information together across recordings so that it can be plotted/analyzed

*getSingleGausFitPlotSigma* Fits single gaussian curve over orientation responses which span 0-180 degrees or test orientation regardless of direction (*PTBOrientationWColorMnky*). Collates the sigma value (the tuning width) and the R2 value (goodness of fit) and adds them to the experiementStructure.

Fixes

*convertExperimentStructure2ClassObject* This converts all experimentStructure files into class objects in line with the new version of code. Should only need to be run once on all old data.

Pixel Based Analysis

*pixelwiseOrientationSelectivity* creates pixel by pixel orientation preference and selectivity images for dataset. If you used *prepDataBatchWithOrientationPixelPref* to preprocess the data then this is done automatically.

*createStimVSPrestimSTDDiffGPU* creates a difference (subtraction) image for both SD and mean for stimulus vs prestimulus time (currently only used for vascular imaging).

Vascular

*lineAnalysisForBloodVessels* creates line profile plots used to analyze vessel diameter for blood flow metrics (used in F32 application)

Plotting

*plotCellOrienationPref* creates a RGB image of maximal response for orientation preference for every cell

*plotOrientationTuningPerCell* plots average condition traces for individual cells

*plotRandomROIFPerCnd* allows the user to choose random ROIs and creates a condition average response plot similar to those produced from *plotOrientationTuningPerCell*. Does not add ROIs or the data to the experimentStructure.

*polarPlotOrientation* creates orientation polar plots for cell traces.

**Data Viewer App**

This code package comes with an interactive data viewer app (“plotting\dataViewerApp”). Once you install the package you can use it to look at the data in any processed experimentStructure.mat. Once you have loaded in the data choose the imaging channel and primary condition no (i.e the number of orientations). On the right hand drop down you can choose the data you want to view. You can then left click on any cell ROI and the data for it should display in the right panel. Additional commands are shown below:

* + - Spacebar- brings up dialog box to choose specific cell number
    - Left/right arrow- display data from the previous cell number or next cell number
    - Up/down arrow- increase and decrease the magnification of the cell ROI image (NB in this mode the view should move to center the currently selected cell).

**Experiment Scripts**

*PTB\_Melanopsin\_Mnky* Interacts with Paul’s Arduino based sheeple LED stimulator to produce Blue/Red LED stimulation of varying brightnesses.

*PTB\_Melanopsin\_Mnky\_Flash* Brief Blue LED flashes of light at highest luminace, used for testing visual response for virus injection guidance during surgery.

*PTB\_RFMappingByHand* Presents a moving gabor which can be changed in size, position, and movement direction by keyboard controls.

*PTBContrastVSOri* Run a contrast and orientation (0-360) sinusoidal grating stimulus set.

*PTBOrientation* Runs an orientation (0-360) sinusoidal grating stimulus set.

*PTBOrientationIntensityMnky* Runs a contrast and orientation (0-180) square wave grating stimulus set.

*PTBOrientationSFMnky* Runs a spatial frequency and orientation (0-180) square wave grating stimulus set.

*PTBOrientationSFMouse* Runs a spatial frequency and orientation (0-360) sinusoidal grating stimulus set.

*PTBOrientationWColorMatchBW\_Mnky* Runs a specific cone isolating color stimulus and the luminance matched monochrome version with orientation (0-180) square wave grating stimulus set*.*

*PTBOrientationWColorMnky* Runs a cone isolating color stimulus with orientation (0-180) square wave grating stimulus set*.*

**Neural Net Training for Automated ROI Selection**

Currently the code package comes with a pretrained convolutional neural net which chooses initial cell ROIs. This net is based on the UNet (Olaf Ronneberger et al 2015, “U-Net: Convolutional Networks for Biomedical Image Segmentation”). This net has been training using the “STD\_Average.tif” images and the “ROIcell.zip files”. Input image size 512 x 512 x 1 at 16bit levels. NB to use this neural net code you must have Matlab 2019 or newer.

If you would like to train your own network for your own data, please follow the steps below:

1. Run *copyDataForNeuralNetTraining* to create a folder with the training data
2. Run *preprocessImages4UNet* to create the images and ROI masks used as the training library
3. Run *createUNetForROIExtraction* to create and train the neural network. Be aware this can take up to a week of solid computations to complete
4. Run *testAccuracyTrainROINet* to test the accuracy of the trained network
5. Move the neural net to the location (‘\extraction\ROIExtraction\’) in the code folder structure to use in *chooseROIs* function.