

# Batch Image Processing

Jeremy Bergsman      [jeremy@bergsman.org](mailto:jeremy@bergsman.org)

## Introduction:

Purpose: This tool is designed for analyzing FM dye destaining, and to a lesser extent, other fluorescence measurements, in cultured neurons, although there may well be other useful applications. You point it at one or more folders, each containing one experiment, and it processes all of them in the same way. The processing consists of subtracting the local background, aligning the images, finding release sites, manual (user) editing of release sites, measurement of the fluorescent signal at each release site through time. The fluorescence change at each release site is optionally assessed for “quality” according to several parameters. A map of the release sites, color coded for quality, is placed over an image of the neurons. Also, the signals of those sites exceeding a quality threshold are averaged and graphed.

Caveats: The current version should be called a beta version, meaning I would not be surprised if it has problems that show up when other people try to use it. It certainly works for me and if nothing else I think it could save someone else a lot of work to build something similar. A couple other people have also made it work. Knowing how many people are doing this type of experiment I offer it for anyone to try. However, support from me may be minimal if you have problems using it. Likewise, this document may be missing some detail—sorry. If you can program have a look at the code—I have tried to comment it pretty well.

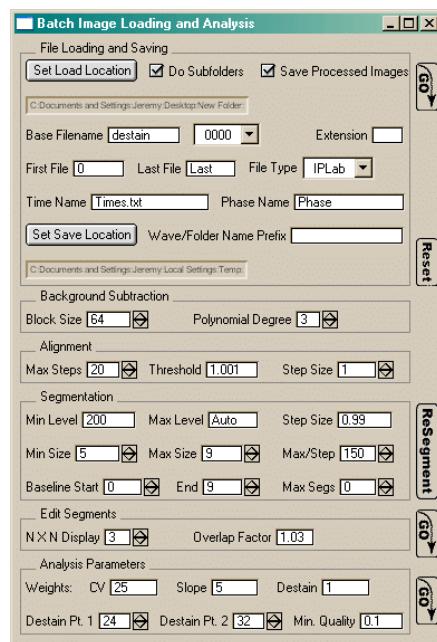
Typography: In this document I have used a **different font** to refer to elements that appear on the screen during use to alert you to look for them there.

Your Data: Each experiment must be contained in its own folder. An experiment consists of a series of fluorescence pictures, one transmitted light picture (phase contrast or DIC or similar), and an optional file containing timing information for the fluorescence pictures. See below for information on file names. The program was written to process IPLab image files, but there is some support for Wasabi RBF (Hammamatsu) image files and TIFF image files.

The timing file should be ASCII (plain text) with one time entry (in seconds) on each line. If the timing file is absent, the program will check for an IPLab timing file in the sequence of images. If this is also absent, the first image will be set to time 0 and remaining images will be assumed to be 1 second apart. If you have multiple experiments to process at once, they should each be in their own folders, with each of these folders in one higher level folder.

## Operation:

To operate, execute the **BatchImageProcessing** macro to call up the panel seen at the right. Set all the parameters as desired (described in detail below) and press the  button to the right of the **File Loading And Saving** box on the panel. (Of the five buttons shown on the right side of the panel, only this one will be visible at first. The others appear after you have run your first



analysis.) All your images will be loaded and processed. After you have run through the process once, new buttons will appear, at which time you can do two types of things: (1) You can load and process a whole new set of experiments. (2) You can reanalyze all or some of the previously processed experiments (presumably after changing some analysis parameters on the panel). Reanalysis can begin at one of three steps (see below for more detail about these steps): segmentation, segment editing, or destaining measurement and quality analysis, by pressing the  or  button next to the appropriate parameter box. The arrows on the  buttons are to remind you that all actions from the arrow, down to the bottom of the panel are executed by a press of the button. The  just resegments without further analysis. By default the most recently loaded set of experiments is reprocessed when any of these buttons is pressed. To adjust which experiments are reprocessed, press the  button. This calls up a list of all previously loaded experiments with check boxes to turn on or off reprocessing. **Reprocessing overwrites previous results for that experiment.**

Note on operation: because Igor files using this tool could conceivably consist of large numbers of very large image files, the images are not held in memory/in the Igor file. Each image in one of your experiments is loaded, processed, and saved. It is then reopened when needed for analysis. If you wish to have a complete record of the processing you must be sure to keep these saved images as they are not associated with the Igor experiment file. Also, reprocessing will look at the current Save Location for saved files and take any that have the right name, so again it is important to keep track of where and whether you save processed images if you will be reanalyzing them.

**File Loading And Saving:** Press the **Set Load Location** button to open a dialog which lets you point at the folder you wish to analyze. Your setting is displayed below the button. You must either point at a folder containing images from one experiment, or at a folder containing subfolders each of which contains one experiment. Check the **Do Subfolders** checkbox if you are analyzing multiple experiments contained in subfolders within the folder you pointed to.

Fluorescence filenames are specified on the next two lines. They consists of an optional alphanumeric part, a required incrementing digit part, and an optional extension. For example, the default setting is for files named “destain0000”, “destain0001”, “destain0002”.... Here, “destain” is the **Base Filename**. You must also indicate the **Number of Incrementing Digits**, which in this case is 4, by selecting “0000” from the popup menu. In this example there is no extension, but if there were, that would be entered in the **Extension** field. On the next line, the **First File** and **Last File** control the range of files processed (the values of the incrementing digits). For **Last File** only you can enter “last” to go until it runs out of files (it won’t skip over a missing file, it will just stop). The **File Type** popup menu lets you specify the image file format (IPLab, Wasabi/RBF, or TIFF, only basic TIFF files are supported).

The **Time Name** is to specify a file that contains timing information. If you have an IPLab timing file as the last file in the fluorescence file naming sequence, it will be recognized and used if no timing file is found. The **Phase Name** should point at a phase contrast or similar picture. Both of these fields should contain the complete file name with any extension. The **Set Save Location** button points at where you want the processed images saved, whether temporarily or permanently. Your setting is displayed below the button. The **Save Processed Images** checkbox controls whether processed images (not your raw images) are deleted from the hard disk after being used for analysis. (By default Igor is not able to delete files from the disk, for images to actually be deleted you must turn this on in the preferences.)

**Background Subtraction:** This works by first dividing the image up into blocks (with size specified in pixels by the **Block Size** field). The dimmest pixel in each block is considered the background and a smooth 2D polynomial (whose degree you set with the **Polynomial Degree** field) is fit to those points. This is then subtracted from the image. Note that this means that you can have negative intensities in the subtracted image.

**Alignment:** This works by finding a particularly bright area in the averaged baseline image (see below) and adjusting every image to maximize its brightness in this area. The adjustment is a series of XY movements of the images, called steps. There is no rotation or subpixel movement. Each step can be allowed to be one or more pixels in size, according to the **Step Size** parameter, but unless you have large shifts from image to image you should keep this set to 1 as it slows processing considerably to increase it. A shifted image must be brighter than the starting image by the amount specified in the **Threshold** field to be accepted. The maximum number of steps allowed is specified in the **Max Steps** parameter. A wave indicating how the images were shifted is created, called Shifts\_Y where Y is your experiment name. If Max Steps is set to 0, no alignment is performed and Shifts\_Y is not created.

**Segmentation:** There are currently two methods for finding “segments”, also known as ROIs, the areas which will subsequently be analyzed for fluorescence changes. Images from your baseline region—specified with the **Baseline Start** and **Baseline End** fields—are first averaged together to provide a lower noise image for segmentation. (When specifying the baseline the values you enter should be the image file numbers. In other words, if your first image is destain0002 and you want the baseline to start with this image, then **Baseline Start** is 2.)

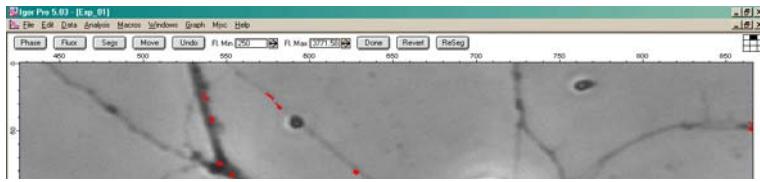
**Iterative Thresholding Method:** This method is based on an algorithm by Stefan Kruger. This attempts to find small segments which are local maxima in intensity and that fit a size criterion. (In my case these are sites of supposed neurotransmitter release and the associated vesicle exo/endocytosis.) The averaged baseline image is scanned from high to low intensity looking for contiguous areas of the right size. You can specify the maximum intensity to consider (to block out areas that are too bright for whatever reason such as saturation) in the **Max Level** field. The intensity steps are to a fraction of the previous intensity, set in the **Step Size** field. The **Min Level** field sets the final intensity of this scan. If you go too low you will have large numbers of spurious segments. Besides keeping this value higher, you may also cause segmentation to stop when a certain number of segments are found at a given intensity—usually a sign that you are at the background. This is the **Max/Step** parameter. You may also specify the maximum total number of segments with the **Max Segs** parameter (leave it set to 0 for no limit). This might be useful to control for bias in segmentation due to intensity by always taking the brightest N segments. Note that setting a limit here will almost double the time required for segmentation. Finally you may set the **Min** and **Max Size** (in pixels) for the spots.

**Single Threshold Method:** (Options not shown on panel example above.) This approach is more suited for larger objects where size and intensity are less homogeneous, and objects are more clearly separated from the background. An intensity histogram of the baseline average is fit with a Gaussian distribution to find the baseline. You specify the number of **Gaussian Widths** above the mean baseline to use as a threshold. Then Igor’s ImageMorphology command is applied. See the Igor manual for more information. In particular, note the impact of the shapes used and the number of iterations, both of which can be specified here for each step: the command is applied first with dilation, to join up neighboring loose points from an object near threshold intensity, and then with erosion, to eliminate spurious points and ensure that the final segments are slightly smaller than the

objects they represent. Accordingly, the shape size and/or number of iterations for the erosion step should be larger than for the dilation step. Finally you can specify the **Min Size** (in number of pixels) to eliminate spurious small segments.

**Information for both methods:** Good values for all of these parameters depend heavily on your images. Note: a region around the border corresponding to the alignment steps is not checked for segments. This is to ensure that you don't analyze "off the edge" of a shifted image. Also, you can have specific pixels ignored (for example if your camera has a bad pixel). Search the code for "//Bad pixel exclusion:" for some examples. See Installation below for controlling which method is used.

**Edit Segments:** Once the experiments have been loaded and processed, you have a chance to remove spurious segments by hand. A portion of the segment editing tool is shown below. It shows the segments in red, superimposed on the phase (black and white) and/or baseline fluorescence (in green) images. Each of these three elements can be toggled on/off with the appropriate button. (Confusingly, if both phase and fluorescence are off they are both displayed.) The minimum and maximum of the displayed



fluorescence range can be adjusted with the appropriate controls to improve the image. (To show both the phase and fluorescence images, a combined image must be computed which may take a few seconds the first time it is needed.)

Clicking and dragging a box with the cursor (the Igor "marquee") on the image deletes any segment pixels within the marquee. The **Undo** button reverses the latest deletion, and the **Revert** button reverses all the deletions to that experiment. The **ReSeg** button resegments just that experiment (presumably you have changed some parameters on the main panel first).

To view sufficient detail, the image may be divided up into an **N X N** grid, showing only one element of the grid at a time. You may set **N** on the main panel. You can also set the fraction of overlap between views of adjacent grids (e.g. 1.03 is 3% overlap). There is a guide in the upper right corner of the segment editing window showing which element you are viewing. In the example above you are viewing the upper middle element of a 3 X 3 grid. When you are done editing an element, press the **Move** button to move to the next element. When you have seen them all the **Done** button will become active, allowing you to move to the next experiment, or to analysis if this is the last experiment. At this point every segment remaining will be measured for each image in the experiment (these are loaded temporarily at this point). A new Igor data folder is created for each experiment, called ResultY where Y is the experiment name. The data folder contains one wave for each segment called ResultsN (where N is a serial number starting from 0), containing its intensity throughout the series of images. Contained in the wave note is an X,Y value of where the segment is in the image, the average value during the baseline, and, if quality analysis feature is turned on, the "Quality", as determined below. These values can easily be extracted programmatically as demonstrated at various places in the code.

**Analysis Parameters and output:** Here you can adjust the various parameters that affect the "quality" score of a segment. The **CV** and **Slope** of the baseline and the amount of fractional **Destaining** (decrease in signal) are combined with the supplied weights to come up with the quality. The destaining is measured at two points (which you need to enter), correcting for any negative baseline

slope (positive slopes are set to 0 for this calculation) and the greater value is used. (When specifying the **Destaining** points, the values you enter should be the image file numbers, like the baseline specification described above.) Intensity curves for segments exceeding the supplied **Min. Quality** are averaged together and displayed. Also the segments are color coded according to quality and superimposed on the phase image producing a “quality map”. If you turn off quality analysis (see Installation below) then all the result waves for each experiment are averaged together and displayed, and no quality map is made.

Another function, “**OneLayerMovie**” is included. If you display one of the saved processed images with an overlay, this will cycle through the saved images while maintaining the overlay. In this way you can create an annotated movie.

### **Installation:**

Your monitor resolution should be something like 1024X768 or higher. 800X640 will cut off the bottom of the main panel. You will probably want to run Igor in full screen mode. You must have Igor Pro 5.03 or later. Installation is similar to any Igor user procedure. Just put a shortcut/alias to the folder containing the “image\_analysis.ipf” file in your Igor Procedures folder. Near the top of the code you will find a few lines you can change depending on your installation:

```
//Various parameters depending on your desired processing:  
Static Constant kThreshSeg=0 //0=Segmentation by iterative thresholding, good for small spots of...  
//1=Segmentation by single threshold followed by erosion and...  
Static StrConstant ksUserMode="JBB" //"Learn" for quality components output, "" for normal...
```

ksUserMode and KThreshSeg are constants that the program looks at to control its behaviour. KThreshSeg sets the segmentation method (see above). ksUserMode has various uses. I have installed a number of features peculiar to my work which are only executed when this constant is set to “JBB”. Usually you will want to leave it null (“”) for full operation. If you set it to “NoQual” then the quality analysis is not performed as described above, and instead an average of all result waves is produced. If you set it to “Learn” then graphs are produced for each segment annotated with the various quality parameters. This lets you consider how to adjust them for your desired weighting.

### **Version Information:**

The panel, once created, is saved with an experiment. If you update to a newer version and reopen an old experiment with a panel from the older version there may be some funny behavior. This can be avoided by closing the panel and reselecting **BatchImageProcessing** from the Macro menu. As the panel relies on the existence of certain waves for reanalysis, this is not guaranteed to work across versions.

### **Copyright Information:**

Programs and manual copyright © 2004-5 This manual was last revised 10-May-05 and refers to version 1.2

The described programs are free software; you can redistribute them and/or modify them under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or any later version. These programs are distributed in the hope that they will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details:

<http://www.gnu.org/licenses/gpl.html>

Free Software Foundation, Inc.  
59 Temple Place, Suite 330  
Boston, MA 02111-1307 USA