I. Related to 3D image reconstruction and analysis

I. 1. 3D object representation, parameterization (geometrical and topological), and arithmetic operations of 3D objects. A 3D object is represented by a collection of connected voxels. Two voxels are considered as connected if they share at list one common vertex. For memory and computational efficiency, the data class represent 3D object is implemented through a hiarchy of data class represent "layer", "patch", and "segment", and record only the boundary voxels (see figure I1 for details). I also implemented two arithmetic operations for 3D objects, i.e. how two 3D objects merge or one exclude another to form a new 3D object. The computation of geometrical properties of 3D objects are also implemented. A 3D object also contains basic topological properties, i.e., whether an object is enclosing other objects or is enclosed by other objects. Such a 3D object representation is a very convenient and efficient representation of neurons, which posses very complicated geometrical and topological parameters.

I. 2. Mouse brain cell distribution simulation program. This module constructs 3D objects representing mouse brain regions according to the three-dimensional coordinates of Allen Reference Atlas structures (Allen Brain Institute), and generates 3D objects representing the brain cells inside specified brain regions according to specified density functions. After implementing more realistic signal and noisy voxel value simulation, it will be used for generating test cases to evaluate the performance of different image alignment and image segmentation programs.

I. 3. A novel statistics-based edge detection algorithm. This is a two-step edge detection algorithm. In the first step, all positions will be scored and those positions scored higher than the cutoff value will be identified as edge candidates. All connected candidates will obtain collective score in the second step. Only those candidates whose collective scores are higher than the cutoff value will be detected as edge.

II. Related to single-molecule fluorescence signal analysis.

II. 1. Statistics based single-molecule detection algorithm. To detect single-molecule signals in an image, a reference image will be created by randomizing the pixel positions of the original image. Landscape analysis will be performed on both the original and the reference images. The local signal height of a region is defined by the mean of the pixel values of the points within the region subtracted by the median of the pixel values of the border points. A quantile (0.01 is the default value) value of the local signal heights in the randomized image is used at the cutoff value for signal detection. All regions in the original image whose local signal heights are higher than the cutoff value are detected as single molecule signals. Such cutoff value is the probability of finding local signal height equal or greater than the cutoff value in an image containing only spatially uncorrelated noise.

II. 2. Automatic fitting of the detected signal regions using multiple component 2D Gaussian functions. The number of components is automatically determined by assessing whether fitting with additional component(s) significantly improve the fitting results.

This is a very challenging task numerically, which we have achieved by our initial fitting parameter estimation algorithm. The reliability of the fitting is important for several reasons. First, it is important for accurate estimation of the pixel values contributed by the molecule of interest, since the accurate fitting made it possible to compensate for the contributions of the signals from other molecules of the region and to adjust the background value of the border points accordingly. Second, it is important for assessing the shape of the signal. It is not unusual to see the cases of signals displaying none spherical shapes. This is caused by multiple molecules located in close proximity or by artifacts. Being able to assess the signal shape is important for automating the detection of signal jumps.

II. 3. Graphical interface of the intensity peak object (IPO) analysis software. Two interfaces are most frequently used for interactive detection of photo bleaching. **A)** the main interface that allows users to run fitting and tracking programs. The main interface also allows users to select and view the parameters of objects and tracks, among many other functions. **B)** the transition detection interface allows users to display the trace of single-molecule signal together with images of the molecule, and many other intermediate images.

III. Common computational modules and graphical interfaces

III. 1. A versatile histogram handling algorithm and subset algebra. This is a versatile utility class. It contains some strategies for automatic determination of bin size, and range. It dynamically updates (loads or removes) data, so it is very convenient and efficient for computing the distributions of values within running windows. It also computes and dynamically updates most common statistics such as mean, standard deviation, and the quantile. Furthermore, it can also be used for clustering objects into subsets by enabling the histogram to store and to retrieve the indexes of the objects whose attributes are used to build the histogram. Therefore, it is also a convenient way to implement a distance based clustering.

III. 2. A library of most commonly used basic operations. To maximize the reusability of the source code, I have built several utility java classes that provide commonly used methods in statistical analysis, i/o operations, GUI implementations, and image processing.

III. 3. A signal intensity landscape analysis program. After segmenting an image using the watershed method, all pixel positions on the image are assigned with one of the following four landscape types, local maxima, local minima, watershed and regular points. All watershed points are connected to divide an image into isolated regions, each of which contains a single local maxima. This landscape analysis is the basis of the single-molecule detection algorithm. It also computes geometric properties and characterizes the topological properties of the regions. This program is also used for many other purposes in the software, such as for finding the optimal initial parameter

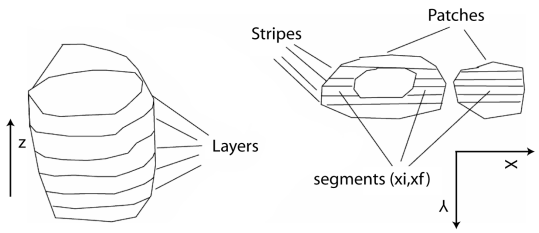
values for automatic fitting of single-molecule signals using multiple component Gaussian functions.

III. 4. Image comparison viewer. I have developed a software with graphical interface to display multiple images side by side in a synchronized manner, while maintaining the same magnification, the same display contrast, and the same selection of the region of interest. I developed this viewer because I needed to intensively compare the original image with the fitting results and many intermediate resulting images. The Image comparison viewer is a very helpful tool in tracking the results of the object tracking algorithm by comparing images of the same area in several consecutive frames.

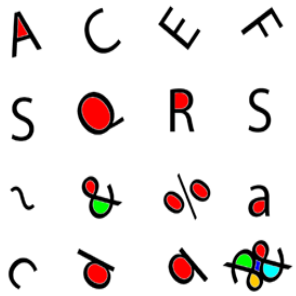
III. 5. Non-linear Image fitter. This module is developed because of our need for a convenient and robust image signal fitting software. I have developed robust parameter initialization methods for several commonly used function types.

III. 6. Plot Handler: I have developed a convenient GUI to handle plots of 2D data. It perform many common analysis of 2D data. It is also very convenient tool for comparing different data set graphically because of the scale synchronization between a group of plots.

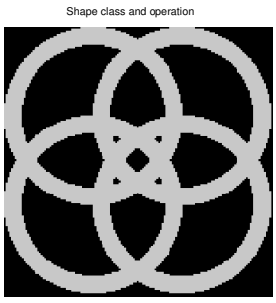
I. 1. Representation of 2D and 3D image objects and operations.



Class ImageShape is used to represent Image objects in images. It contain data members that record the boundary of the objects and methods that implements the calculation of geometric properties (such as area, perimeters), characterize its topological properties (such as connectivity). It also contains methods for common operations that create new shapes from existing shapes such is merging, overlapping and excluding. A 3D image object in a image stack is represented by class ImageObject that contains data member recording ImageShape object identified each slice, as well as methods of geometrical and topological characterizations. It also contain methods for common operation to create new shapes from existing ones.

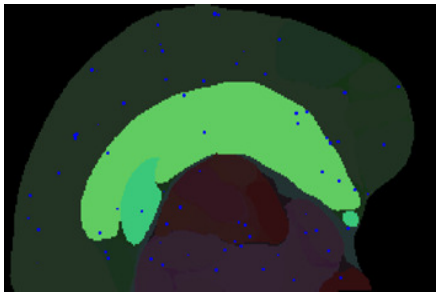


Example of Identification of other objects enclosed in the main objects. The program identified each dark object, defined by connected points with lower than cutoff pixel values, then identified other objects enclosed within the main objects. In this case, the program identified the white areas within each dark objects and colored each of them with distinct color.



Example of shape operation. The image is output of a new shape obtained by merging of five rings.

I. 2. Mouse brain cell distribution simulation program. This program reads in the three dimensional coordinates of Allen Reference Atlas structures (Allen Brain Institute), constructs object representation of each brain structure, generate simple 3D geometric structures (such as ellipsoids and cylinders) according to the given densities assigned to specific brain regions in the brain atlas structure. This module is developed for customized parameterization of the structures of mouse brain regions. Since it can generate images of the brain structure (together with the enclosed geometric objects) sectioned at arbitrary depth and angle, the generated images can be also used as the test cases to assess the accuracy of the stitching and alignment programs.



Example of an output of cross section image after creating ellipsoids in the mouse brain atlas structure. A uniform cell density was assigned to the entire atlas structure.

I. 3. Development of a novel edge detection algorithm. Edge detection is one of the key tasks in digital image processing, because successfully detected edges convey information about location, geometric features and textural features of objects. Edge detection is also an important initial step for subsequent tasks in image processing. Therefore edge detection is an area that has attracted extensive research in the field of image processing (e. g. William K. Pratt, 2007). Edge detection approaches can be categorized into two types, algorithms based on the local pixel value derivative (first order or second order derivatives) and algorithms based on statistical tests (e. g. Lim, 2006).

One commonality among all existent edge detection algorithms is that the scoring system (derivative based or statistical test based) does not take into account how a pixel point being scored as a possible edge is related with the scores of its neighboring points. In reality, edges of objects in real life should be considered as a line segment of continuously connected points. Therefore we argued that it should be more reasonable to use an edge detection scoring system in which the score of one pixel point is related with that of its neighboring points.

To implement such an edge detection scoring system, we have developed an algorithm that evaluates a point for the likelihood that it forms a continuous edge together with its neighboring points, instead of evaluating a point being an edge point by itself. It is a 2-step process: First, the possibility of a point being an edge point will be evaluated similar to other statistics-based edge detection algorithms. Second, if the score passes a predefined initial threshold, then it will be compared with its neighboring points located along the direction of the initial hypothetical edge. If the initial scores of the neighboring points also pass the initial threshold and if they also have the same edge directions, then a final score will be evaluated for the combined hypothetical edge formed by all neighboring points. The final score is the p value of the comparison of the pixel points on two sides of the combined hypothetical edge. If the p value is lower than the predefined threshold value for the final score thus indicating significance, then all neighboring points will be identified as edge points. The initial threshold is substantially lower than the threshold used in other statistics-based edge detections, so potential edge points will not be excluded as easily based on the initial score alone. The final threshold is set much higher so there is little chance that the random alteration of the pixel values by the superimposed noise will be detected as edge elements.

Figure 1 shows the performance of our edge detection algorithm on a synthetic image, when the image was corrupted by bringing the noise level (σ_n=30, 8-bit gray scale level) up to a value comparable to the height of the signal (h=40). Up to such a high noise level, the edges detected by my algorithm are naturally connected along the boundary of the circle. At this noise level, the trace of the circle boundary based on the Sobel detector is not possible to recognize even by the human eye. We have shown the comparison with the Sebel detector because it is commonly used in the literature as the reference, and also because it is the built-in edge detector in ImageJ (NIH), the platform we used to develop our edge detection program. We also compared the performance of our program with

more sophisticated edge detectors including the Canny-Deriche detector, thanks to Thomas Boudier who made the ImageJ plugin available to general ImageJ users. At the noise level of σ_n=30, although the trace of the circle is recognizable from visual inspection based on the edges detected by the Canny-Deriche detector, there are numerous false positive edges detected throughout the image and a substantial portion of the true edges failed to be detected as edges due to the alteration of the pixel values by the noise (not shown). To the best of our knowledge, there are no reports of edge detectors that perform successfully at such a low signal to noise ratio as ours did.

The key factor of the successful performance of our edge detection algorithm up to such high noise levels is in that it utilizes the most basic characteristics of noise, the randomness, to keep the noise from being detected as edges. It is quite possible that the altered pixel values due to background noise may cause a high score for a given pixel point, but it is much less likely that the apparent edges created by noise would have a consistent direction for neighboring points, as expected for a true edge. Although the susceptibility of the edge detector to high noise levels can be reduced by increasing the window size of the detector, such approaches lose the sensitivity and precision for edge detection. Our approach is different in that it evaluates the combined scores only among the initially detected edge points displaying a consistent direction, so it minimizes the chance that the scores and the locations of the true edge points will be affected by other points included into the consideration due to the predefined large window size.

William K. Pratt, “Digital Image Processing: PIKS Scientific Inside”, 2007.

Dong Hoon Lim, 2006 “Robust rank-order test for edge detection in noisy images”, Journal of Nonparametric Statistics, Volume 18, pages 333 - 342

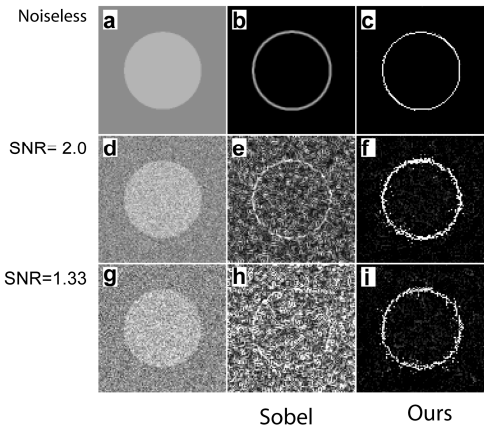
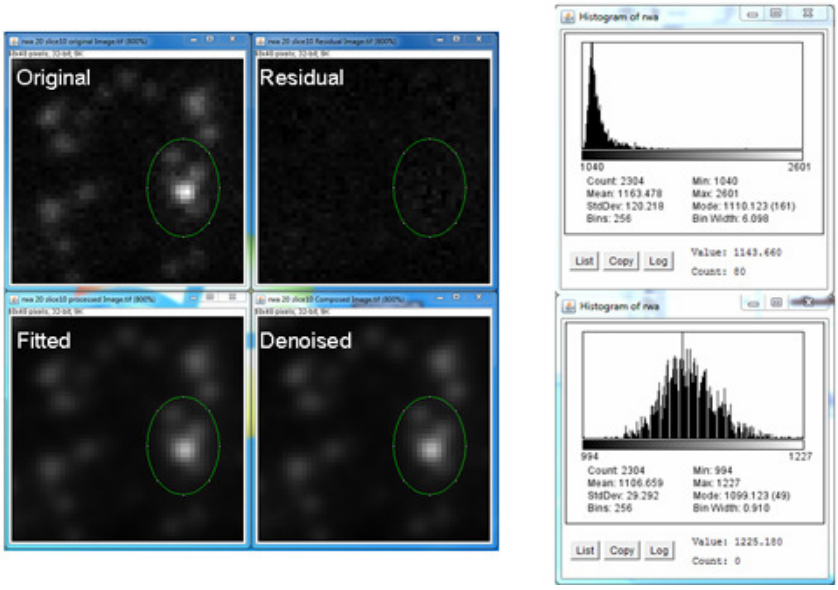


Figure 1. The performance of our edge detection program in synthesized images corrupted by various levels of noise. The original image (a) is a circle with a uniform pixel value of 180 (8-bit gray level value) on the uniform background with a pixel value 140, giving the height of the net signal as 40 pixel values. Both of the edge detectors performed accurately in the absence of noise (b and c). When the image was corrupted by Gaussian noise of standard deviation being 20 pixel values, σ_n=20 (d), an excessive number of points are marked by the Sobel detector with edge scores similar to those of points located around the boundary of the circle (e). Although it is possible to recognize the existence of the circle by visual inspection, it is not very feasible for computer programs to connect the edge points around the boundary of the circle (e). When the image was corrupted by Gaussian noise of standard deviation being 30 pixel values, σ_n=30 (g), the existence of the circle was not recognizable even by visual inspection based on the edge points detected by the Sobel detector (h). In contrast, the edge points detected by our program can be naturally connected along the circle boundary at both moderate and high levels of noise (f and i).

II. 3. Automatic fitting of the detected signal regions using multiple component 2D Gaussian functions. The number of components is automatically determined based on whether fitting with additional component significantly improve the fitting results. This is a numerically challenging task, which we have achieved by our initial fitting parameter estimation algorithm. The reliable fitting is important for several purposes. Firstly, it is important for accurate estimation of the pixel values contributed by the molecule of interest. The accurate fitting made it possible to compensate the contributions of the other molecules to the pixel values of the region and to the background value of the border points. The accurate fitting also made it possible to parameterize the shape of the signals.

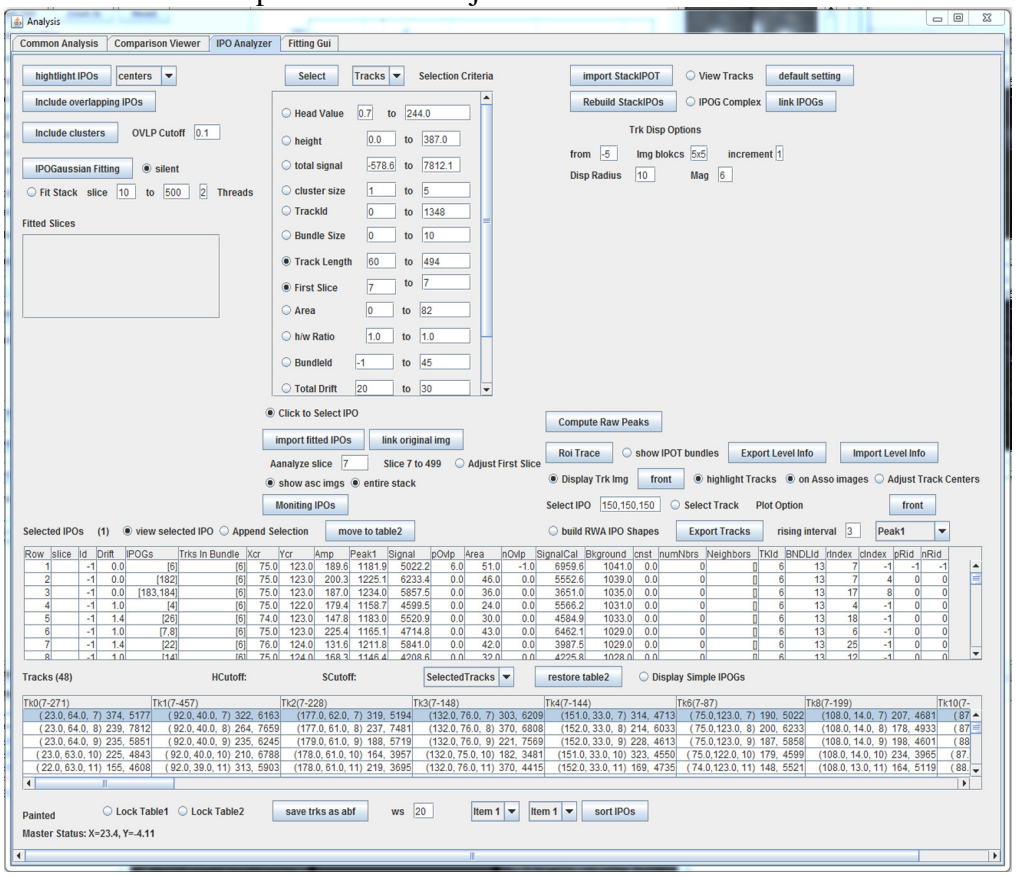
Automatic fitting results of Single-molecule fluorescence signal Viewed by comparison viewer



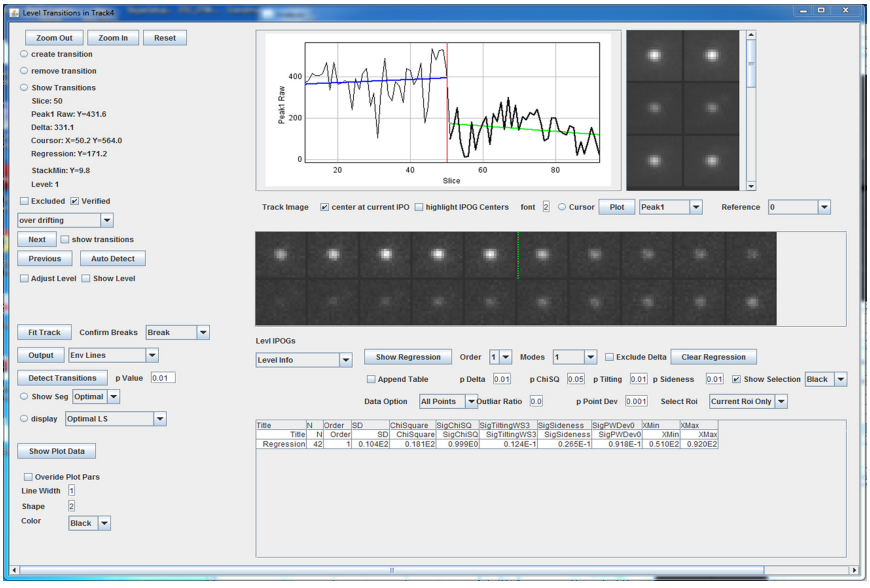
This is an example of automatic signal fitting by Gaussian functions. The left panel of the above image shows the comparison of four images, the original image, the fitted image, the denoised image, and the residue image. The denoised image is obtained by filtering the original image using Gaussian blurring filter (radius=1). The residual image is obtained by subtracting the fitted signal values from the original image. The satisfactory performance of the fitting software can be appreciated not only by visual comparison of the fitted and denoised image, but also by the absence of visible signals in the residual image.

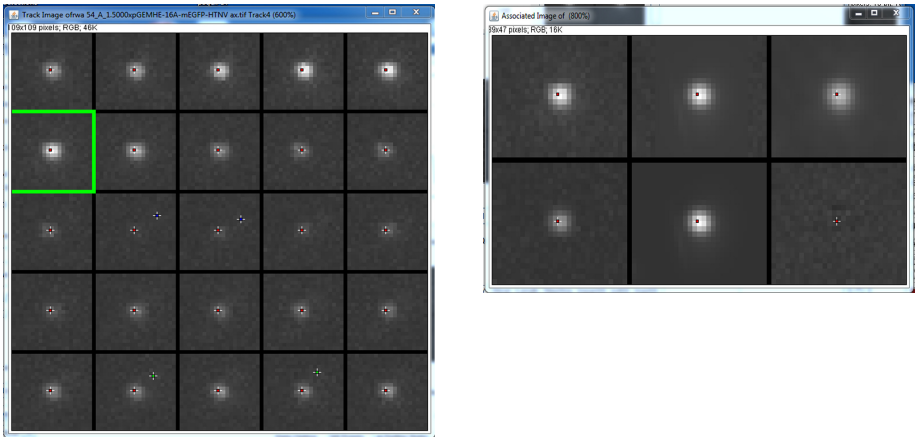
The satisfactory fitting results can be also appreciated by comparisons of the pixel value distributions of the original (right, upper panel) and the residual (right, lower panel) image. The pixel value distribution is positively skewed due to the signal values, with minimum, mean, and maximum pixel values as 1040, 1163 and 2601, respectively. The pixel distribution of the residual image is a normal distribution according to a Shapiro-Wilk normality test, an as can be readily agreed by visual examination. The minimum, mean and maxima pixel value of the residue image are 994, 1106 and 1227, respectively.

II. 5 A). The main IPO analysis interface. The main IPO analysis interface is used for fitting and tracking the single-molecule signals. The main interface also allow users to select and view the parameters of objects and tracks.



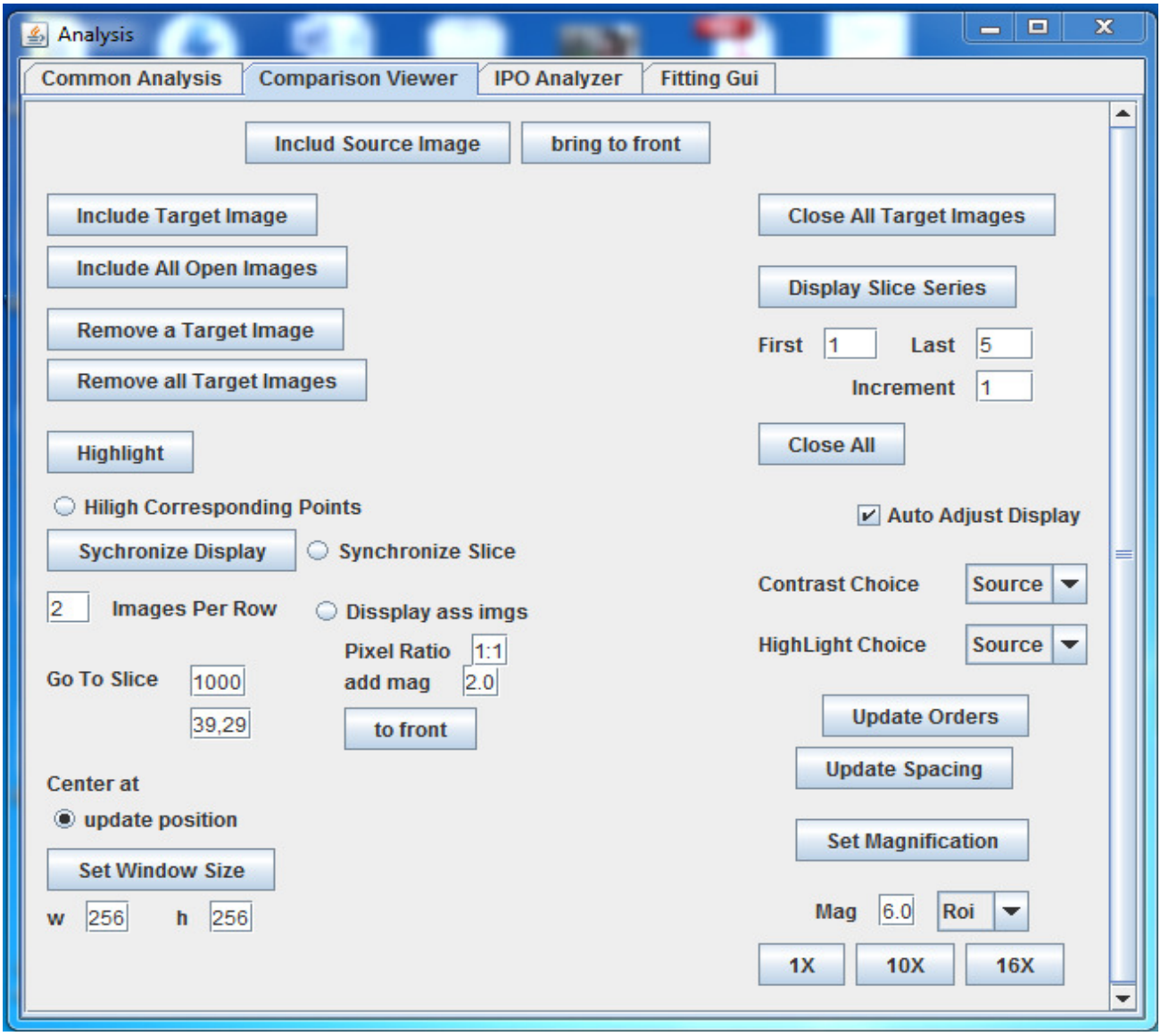
II. 5 B). The transition detection interface. This interface allows users to semi-automatically (automatic detections followed by manual corrections) detect the transition steps.





The user can simultaneously exams signal plat and the signal image. The red vertical line in the plot is the cursor, and the numerical values of the signal are displayed on the space left to the plot. As moving the cursor through the signal trace, the image on the lower panel of the main interface update accordingly, enable the users to decide based on both signal value and the visual inspection of the images. It also prevent any false detection that caused by possible tracking, or fitting errors. The image right to the green vertical line on the image panel of the main interface correspond to the red vertical line on the plot. The lower panels are signal image shown in larger view (left), and the images related to the fitting results (right). The absence of any visible signals in the residual image (lower right panel) indicates the fitted value accurately account for the original signal. These two images are also automatically updated as users browsing through different points on the signal trace. The blue and green regression lines on the signal trace indicates automatic transition detection results. The automatic detection of current version makes some mistakes for the traces with larger fluctuations.

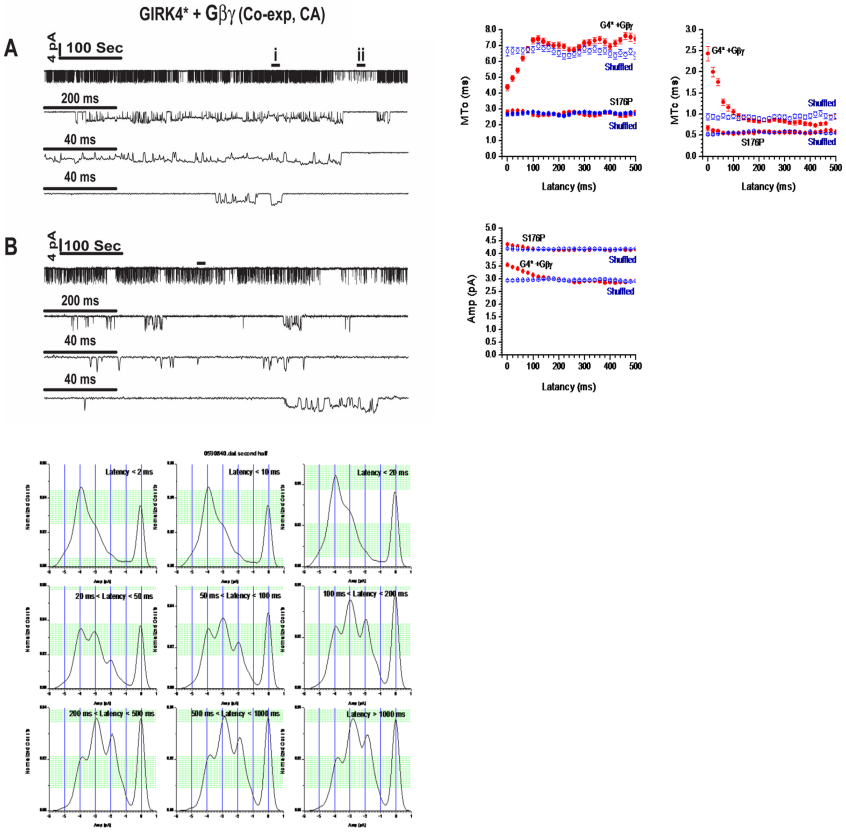
II. 4. Image comparison viewer. I have developed this module to display multiple images side by side using synchronized manner, the same magnification, the same display contrast, and the same selection of the region of interest. I developed it because I need to compare the original image, the fitting results, and many intermediate-result images when I was developing automatic signal fitting algorithm.



When synchronize the display of multiple images, one of them need to be selected as the source image, and the other images should be selected as target images. Above is the image of the comparison viewer's interface. The view can be numerically set by setting the appropriate field values in the interface. The comparison of the fitting results in the figure II. 3 is generated by this viewer.

I. 3. A versatile histogram and subset handling algorism.

This is a versatile utility class. It contain some strategies for automatic determination of bin size, and range. It dynamically update (load or remove) data, so it is very convenient and efficient for computing the distributions of values within running windows. It also compute and dynamically update most common statistics such as mean, standard deviation, and the quantile. Furthermore, it can be also used for clustering objects into subsets by enabling the histogram to store and to retrieve the indexes of the objects whose attributes are used to build the histogram. Therefore, it is also a convenient way to implement a distance based clustering.



Here is an application of this class for single-channel kinetics study. The purpose is to test whether some single channel kinetics parameters such as Mean open time (MTo), mean closed time (MTc) and the unitary current amplitude (Amp) gradually evolve as functions of the latency between the start of the bursts and the channel openings within the burst.

For this purpose, A series of subsets of the single-channel events within bursts are made according to their latency values. Then the program calculated the mean value of the studied single channel parameters of each subset and displayed them as the function of latency. To study the change of distributions of amplitude as a function of latency, we compared amplitude histogram of each subset. The results clearly show that the studied single-channel parameter of the channel did evolve as function of latency. In contrast, the

single-channel parameters of the mutant channel whose single-channel parameters were not expected to vary as the latency did not display the latency dependency.